DOCUMENT RESUME

ED 074 787 EM 011 001

AUTHOR	Levine, David R.
TITLE	Computer-Based Analytic Grading for German Grammar Instruction. Psychology and Education Series. Technical Report Number 199.
INSTITUTION	Stanford Univ., Calif. Dept. of Computer Science.; Stanford Univ., Calif. Inst. for Mathematical Studies in Social Science.
SPONS AGENCY	National Science Foundation, Washington, D.C.
REPORT NO	STAN-CS-73-343; TR-199
PUB DATE	16 Mar 73
NOTE	214p.
EDRS PRICE	MF-$0.65 HC-$9.87
DESCRIPTORS	*Computer Assisted Instruction; *Computer Programs; German; *Grading; *Language Instruction

ABSTRACT

A computer program is described here which will do an in-depth grading analysis of a student's attempt to produce a required German sentence. The grading program is given a description of the expected response and the sentence actually constructed by the student, the two are compared, and a complete diagnostic report of the correctness of the grammar of the student's response is provided for the teacher. The primary limitations on the program at this point are that it handles only grammar and not meaning, it can process only a subset of the German language, and the grading program has only been operated in isolation thus far and has not been included in a teaching environment. (Author/RH)

COMPUTER-BASED ANALYTIC GRADING FOR GERMAN GRAMMAR INSTRUCTION

BY

DAVID R. LEVINE

TECHNICAL REPORT NO. 199
MARCH 16, 1973

PSYCHOLOGY AND EDUCATION SERIES

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

and

COMPUTER SCIENCE DEPARTMENT REPORT NUMBER STAN-CS-73-343

STANFORD UNIVERSITY
STANFORD, CALIFORNIA

TECHNICAL REPORTS

PSYCHOLOGY SERIES

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

(Place of publication shown in parentheses; if published title is different from title of Technical Report, this is also shown in parentheses.)

(For reports no. 1 - 44, see Technical Report no. 125.)

50 R. C. Atkinson and R. C. Calfee. Mathematical learning theory. January 2, 1963. (In B. B. Wolman (Ed.), Scientific Psychology. New York: Basic Books, Inc., 1965. Pp. 254-275)

51 P. Suppes, E. Crothers, and R. Weir. Application of mathematical learning theory and linguistic analysis to vowel phoneme matching in Russian words. December 28, 1962.

52 R. C. Atkinson, R. Calfee, C. Sommer, W. Jeffrey and R. Shoemaker. A test of three models for stimulus compounding with children. January 29, 1963. (J. exp. Psychol., 1964, 67, 52-58)

53 E. Crothers. General Markov models for learning with inter-trial forgetting. April 8, 1963.

54 J. L. Myers and R. C. Atkinson. Choice behavior and reward structure. May 24, 1963. (Journal math. Psychol., 1964, 1, 170-203)

55 R. E. Robinson. A set-theoretical approach to empirical meaningfulness of measurement statements. June 10, 1963.

56 E. Crothers, R. Weir and P. Palmer. The role of transcription in the learning of the orthographic representations of Russian sounds. June 17, 1963.

57 P. Suppes. Problems of optimization in learning a list of simple items. July 22, 1963. (In Maynard W. Shelly, II and Glenn L. Bryan (Eds.), Human Judgments and Optimality. New York: Wiley. 1964. Pp. 116-126)

58 R. C. Atkinson and E. J. Crothers. Theoretical note: all-or-none learning and intertrial forgetting. July 24, 1963.

59 R. C. Calfee. Long-term behavior of rats under probabilistic reinforcement schedules. October 1, 1963.

60 R. C. Atkinson and E. J. Crothers. Tests of acquisition and retention, axioms for paired-associate learning. October 25, 1963. (A comparison of paired-associate learning models having different acquisition and retention axioms, J. math. Psychol., 1964, 1, 285-315)

61 W. J. McGill and J. Gibbon. The general-gamma distribution and reaction times. November 20, 1963. (J. math. Psychol., 1965, 2, 1-18)

62 M. F. Norman. Incremental learning on random trials. December 9, 1963. (J. math. Psychol., 1964, 1, 336-351)

63 P. Suppes. The development of mathematical concepts in children. February 25, 1964. (On the behavioral foundations of mathematical concepts. Monographs of the Society for Research in Child Development, 1965, 30, 60-96)

64 P. Suppes. Mathematical concept formation in children. April 10, 1964. (Amer. Psychologist, 1966, 21, 139-150)

65 R. C. Calfee, R. C. Atkinson, and T. Shelton, Jr. Mathematical models for verbal learning. August 21, 1964. (In N. Wiener and J. P. Schoda (Eds.), Cybernetics of the Nervous System: Progress in Brain Research. Amsterdam, The Netherlands: Elsevier Publishing Co., 1965. Pp. 333-349)

66 L. Keller, M. Cole, C. J. Burke, and W. K. Estes. Paired associate learning with differential rewards. August 20, 1964. (Reward and information values of trial outcomes in paired associate learning. (Psychol. Monogr., 1965, 79, 1-21)

67 M. F. Norman. A probabilistic model for free-responding. December 14, 1964.

68 W. K. Estes and H. A. Taylor. Visual detection in relation to display size and redundancy of critical elements. January 25, 1965, Revised 7-1-65. (Perception and Psychophysics, 1966, 1, 9-16)

69 P. Suppes and J. Donio. Foundations of stimulus-sampling theory for continuous-time processes. February 9, 1965. (J. math. Psychol., 1967, 4, 202-225)

70 R. C. Atkinson and R. A. Kinchla. A learning model for forced-choice detection experiments. February 10, 1965. (Br. J. math stat. Psychol., 1965, 18, 184-206)

71 E. J. Crothers. Presentation orders for items from different categories. March 10, 1965.

72 P. Suppes, G. Groen, and M. Schlag-Rey. Some models for response latency in paired-associates learning. May 5, 1965. (J. math. Psychol., 1966, 3, 99-128)

73 M. V. Levine. The generalization function in the probability learning experiment. June 3, 1965.

74 D. Hansen and T. S. Rodgers. An exploration of psycholinguistic units in initial reading. July 6, 1965.

75 B. C. Arnold. A correlated urn-scheme for a continuum of responses. July 20, 1965.

76 C. Izawa and W. K. Estes. Reinforcement-test sequences in paired-associate learning. August 1, 1965. (Psychol. Reports, 1966, 18, 879-919)

77 S. L. Blehart. Pattern discrimination learning with Rhesus monkeys. September 1, 1965. (Psychol. Reports, 1966, 19, 311-324)

78 J. L. Phillips and R. C. Atkinson. The effects of display size on short-term memory. August 31, 1965.

79 R. C. Atkinson and R. M. Shiffrin. Mathematical models for memory and learning. September 20, 1965.

80 P. Suppes. The psychological foundations of mathematics. October 25, 1965. (Colloques Internationaux du Centre National de la Recherche Scientifique. Editions du Centre National de la Recherche Scientifique. Paris: 1967. Pp. 213-242)

81 P. Suppes. Computer-assisted instruction in the schools: potentialities, problems, prospects. October 29, 1965.

82 R. A. Kinchla, J. Townsend, J. Yellott, Jr., and R. C. Atkinson. Influence of correlated visual cues on auditory signal detection. November 2, 1965. (Perception and Psychophysics, 1966, 1, 67-73)

83 P. Suppes, M. Jerman, and G. Groen. Arithmetic drills and review on a computer-based teletype. November 5, 1965. (Arithmetic Teacher, April 1966, 303-309.

84 P. Suppes and L. Hyman. Concept learning with non-verbal geometrical stimuli. November 15, 1968.

85 P. Holland. A variation on the minimum chi-square test. (J. math. Psychol., 1967, 3, 377-413).

86 P. Suppes. Accelerated program in elementary-school mathematics -- the second year. November 22, 1965. (Psychology in the Schools, 1966, 3, 294-307)

87 P. Lorenzen and F. Binford. Logic as a dialogical game. November 29, 1965.

88 L. Keller, W. J. Thomson, J. R. Tweedy, and R. C. Atkinson. The effects of reinforcement interval on the acquisition of paired-associate responses. December 10, 1965. (J. exp. Psychol., 1967, 73, 268-277)

89 J. I. Yellott, Jr. Some effects on noncontingent success in human probability learning. December 15, 1965.

90 P. Suppes and G. Groen. Some counting models for first-grade performance data on simple addition facts. January 14, 1966. (In J. M. Scandura (Ed.), Research in Mathematics Education. Washington, D. C.: NCTM, 1967. Pp. 35-43.

91 P. Suppes. Information processing and choice behavior. January 31, 1966.

92 G. Groen and R. C. Atkinson. Models for optimizing the learning process. February 11, 1966. (Psychol. Bulletin, 1966, 66, 309-320)

93 R. C. Atkinson and D. Hansen. Computer-assisted instruction in initial reading: Stanford project. March 17, 1966. (Reading Research Quarterly, 1966, 2, 5-25)

94 P. Suppes. Probabilistic inference and the concept of total evidence. March 23, 1966. (In J. Hintikka and P. Suppes (Eds.), Aspects of Inductive Logic. Amsterdam: North-Holland Publishing Co., 1966. Pp. 49-65.

95 P. Suppes. The axiomatic method in high-school mathematics. April 12, 1966. (The Role of Axiomatics and Problem Solving in Mathematics. The Conference Board of the Mathematical Sciences, Washington, D. C. Ginn and Co., 1966. Pp. 69-76.

(Continued on inside back cover)

COMPUTER-BASED ANALYTIC GRADING FOR GERMAN GRAMMAR INSTRUCTION

by

David R. Levine

TECHNICAL REPORT NO. 199

March 16, 1973

PSYCHOLOGY AND EDUCATION SERIES

INSTITUTE FOR MATHEMATICAL STUDIES IN THE SOCIAL SCIENCES

and

COMPUTER SCIENCE DEPARTMENT REPORT NUMBER STAN-CS-73-343

STANFORD UNIVERSITY

STANFORD, CALIFORNIA

Table of Contents

Acknowledgments

PART ONE: Introduction and Literature Review

1. Introduction 3
   Operation 4

2. Literature Review 13

PART TWO: Matching the Student Sentence with the Expected Words and Structures

3. Basic Method 27
   Word Recognition 32

      Lexical Words 35
         Adjectives 35
         Nouns 35
         Verbs 36

      Function Words 38
         Prepositions 38
         Articles 39

      Pronouns 40

4. Implications of the Limiting Assumptions 41
      Boundary 41
      Limitation: Theoretic 43
      Limitation: Education 44

5. Program Architecture 47
      Operation of Phase One 48

      Phase Two 53
      Data Structure 59
         Phase Two Summary 61

      Phase Three 61
         Phase Three Summary 65

      Phase Four 66

ACKNOWLEDGMENTS

This dissertation has been motivated largely through my involvement with the ongoing research in Computer-Assisted Instruction at the Institute for Mathematical Studies in the Social Sciences, at Stanford. I am extremely grateful to my many colleagues at the Institute who have provided patient ears, thoughtful suggestions, and timely assistance; and to Prof. Patrick Suppes, for extending both his personal encouragement and the support of the Institute.

I should also like to thank the many members of the Stanford faculty who have aided my research: Prof. William McKeeman and Prof. Jerome Feldman, my principal advisors; Prof. Joseph Van Campen and Prof. Roger Schank, members of the reading committee; Prof. Dora Schulz and Mrs. Gertrude Mahrholz, of the German Dept., and Prof. W.H. Cowley, for much unofficial assistance.

I was fortunate on a number of occasions to receive advice and encouragement from friends and colleagues outside of the Stanford community.

The personal support rendered by my wife, Regina, provided the essential ingredient for the completion of this dissertation.

This research was partially supported by a National Science Foundation Grant for Basic Research in Computer Assisted Instruction, NSF GJ-443.

PART ONE

Introduction and Literature Review

Chapter One
INTRODUCTION

This dissertation is motivated by a concern for an essential aspect of teaching, particularly of teaching by computer: the necessity of grading the student's response to a question posed by the teacher. By grading, I mean analyzing for correctness, diagnosing what has been done both correctly and incorrectly. My work concentrates on language teaching, specifically the teaching of German grammar, and works at analyzing sentence-sized responses. In the restricted area of pattern practice and controlled translation, in which the general form and content of the sentence are specified as part of the rules of the game, I have produced a computer program which can do an in-depth grading analysis. I will not presume to claim to replace a human teacher, but I do assert that a substantial step has been taken towards duplicating many of the strong points of a good teacher's grammatical analysis.

A few caveats:

1. My work deals primarily with the grammar of a language, not language as a means of communicating ideas. It is intended for an environment where a sentence like "I ate the books" may represent a less serious error than "I ate them eggs."

2. This research does not represent an exhaustive treatment of the German language. It is rather intended as a demonstration of how grading might be done; to that end some features are handled very extensively, while others have so far received very little attention.

3. The grading program is intended for operation in a teaching environment, of course. I have not yet constructed such an environment. I merely assume that the student will be asked, in some way, to produce a specific German sentence, and worry about its analysis.

Operation:

The grading program is given two inputs, a description of the expected response and the sentence actually constructed by the student. Using its built-in knowledge of German grammar, it analyzes the student sentence for conformity with both the expected response and the applicable rules of German grammar, and produces a comprehensive diagnostic report. (This report is intended for a teacher, not directly for the student.) An example of the program in operation may be helpful at this point.

The expected sentence is "Jetzt will er dem Maedchen die Tuer aufmachen" ("Now he will open the door for the girl"). In the first example, the student has responded correctly. In the second, a large number of errors are present, including a change to a first person subject. I have appended a few explanatory notes, set off in square brackets, to facilitate understanding what is going on; a fuller description can be found in several suceeding sections of this dissertation.

——————— RUN 1 ———————

```
Description generated by human expert:
(STMT (VERB WOLLEN (AUF . MACHEN))
      (SJ ER)
      (OJ ACC DIE TUER)
      (OJ DAT DEM MAEDCHEN)
      (PREDMOD (ADVERB JETZT)))
                         [This is the description of the expected
                         sentence, containing both its words and
                         structure.  To wit, a statement (STMT)
                         with two verbs ("wollen" & "aufmachen");
                         subject a pronoun "er" (SJ ER);  an
                         accusative object (OJ ACC) consisting of
                         the noun phrase "die Tuer";  a dative
                         object; and finally, a predicate-modifier
                         in the form of the adverb "jetzt".]

Sentence input by student  ..
JETZT WILL ER DEM MAEDCHEN DIE TUER AUFMACHEN
                         [The student's actual input, which will
                         be the subject of the following analysis.]

NOW PROCEEDING WITH ANALYSIS

                         [The first report will be from the
                         inflectional analysis of the dative
                         object "dem Maedchen".]

..........  SUMMARY REPORT FOR NOUN PHRASE INFLECTION
                ON DEM MAEDCHEN
        Best we can do guessing intent is: (DAT S (M N))
           CONFIDENCE LEVEL: VERYHIGH
                         [The inflectional analysis looks at what
                         the student actually did and tries to
                         find an interpretation for the particular
                         inflectional forms...in this case, it
                         feels that the pair "dem Maedchen"
                         reflects an inflection for dative,
                         singular, masculine or neuter.]
```

.... GENDER REPORT FOR NOUN: MAEDCHEN :: WEAK CORRECT
EXPECTED N , COULD BE (M N)
[A report comparing the above conclusion with what is expected or correct.]

.... CASE REPORT FOR OBJECT: DEN MAEDCHEN ::
STRONG CORRECT = DAT

.... SUMMARY REPORT FOR NOUN PHRASE INFLECTION ON DIE TUER
Best we can do guessing intent is: ((NOM ACC) S F)
CONFIDENCE LEVEL: VERYHIGH
[The computer's informed guess, as before. The "confidence level" refers to the guess, not to the state of the student's knowledge.]

.... GENDER REPORT FOR NOUN: TUER ::
STRONG CORRECT = F

.... CASE REPORT FOR OBJECT: DIE TUER :: WEAK CORRECT
EXPECTED ACC , COULD BE (NOM ACC)

.... CASE REPORT FOR SUBJECT: ER ::
STRONG CORRECT = NOM

.... VERB ENDING CHECK .. OK!
[This is a check on the form "will" of the main (or "finite") verb "wollen". It appears this late because the program had to first find the subject before it could check subject-verb agreement.]

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: VERYHIGH
[This is the basic word-order check reporting on its findings. "FV-2" means Finite Verb-Second, the required pattern for a statement in German. "Confidence" again is an internal performance assessment.]

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH

.... Summary Comments
Clause type correct (FV-2)

——————— END OF RUN ———————

———— RUN 2 ————

```
Description generated by human expert:
(STMT (VERB WOLLEN (AUF . MACHEN))
      (SJ ER)
      (OJ ACC DIE TUER)
      (OJ DAT DEM MAEDCHEN)
      (PREDMOD (ADVERB JETZT)))
                    [Description is the same as for run 1.]

Sentence input by student ..
JETZT ICH WILL MACHN DEN TUER AUF DER MAEDCHEN
                    [The student sentence is quite different
                    for this run. It represents an attempt
                    at the same response, but with many
                    errors.]

NOW PROCEEDING WITH ANALYSIS

.......... VERB MACHEN .. APPEARS AS  MACHN
        MISSING -E ON INFINITIVE ENDING
                    [The stem is "mach-", and since an
                    infinitive is required it should appear
                    as "machen." There is no interpretation
                    under which the "-n" is correct, so the
                    program complains about improper formation
                    of the expected infinitive.]

......... SUMMARY REPORT FOR NOUN PHRASE INFLECTION ON DEN TUER
        Best we can do guessing intent is: (ACC S M)
           CONFIDENCE LEVEL: HIGH
                    [Note that the confidence level is only
                    "high" ... the construction is not
                    correct, as "tuer" is a feminine noun.
                    The program has a hypothesis about what
                    the student intends; this is quite close
                    to the expected values, so the program
                    is reasonably confident of its guess.]

.......... GENDER REPORT FOR NOUN: TUER :: INCORRECT
        Should be F is M
                    [Based on the guess above.]

.......... CASE REPORT FOR OBJECT: DEN TUER ::
        STRONG CORRECT = ACC
                    [Again based on the hypothesis on what
                    the student intended. Notice that the
                    program has thus been able to establish
                    correct performance for case and identify
                    specifically an error in gender.]
```

.... SUMMARY REPORT FOR NOUN PHRASE INFLECTION
ON DEF
Best we can ... ntent is: (DAT S F)
CONFIDENC... :
[Ve... iar comments apply here.]

.... GENDER REPORT FOR NOUN: MAEDCHEN :: INCORRECT
Should be N is F

.... CASE REPORT FOR OBJECT: DER MAEDCHEN ::
STRONG CORRECT = DAT

.... PN SUBSTITUTION.. Should be (3 NOM S M PER)
is ((WORD ICH (I C H)) : (1 NOM S * PER))
[At this point the program has sucessfully coped with an error which resulted in a totally different form, the change from the expected "er" to "ich". The message gives the internal representation for the inflectional characteristics associated with the two forms, from which one can see that the only change is in person, from third to first.]

.... CASE REPORT FOR SUBJECT: ICH ::
STRONG CORRECT = NOM.

.... VERB ENDING CHECK .. OK!
[Even though the subject was not the expected one, the program can check verb agreement with what actually was present.]

.... Phrase division done, sentence type report:
No confidence in sentence type, probably confused
last straw was: Separable-prefix confusion
[This sentence is pretty badly mangled, and the word-order check finally gives up trying to match it to a German pattern.]

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
[This is a standard beginning for the word-order analysis of a sentence: start in good faith, reduce the level of confidence as errors are observed. "FVT" refers to Finite Verb Type, the classification of the sentence (FV-2).]

```
Due to: DI not separated
        Confidence in FVT setting is: HIGH
           [Oops, the dependent infinitive (DI) is
           positioned immediately after the finite
           verb, rather than where it belongs at
           the end of the clause.  Also, its
           separable prefix "auf" has been detached,
           an error for an infinitive;  more on that
           later on.]
Due to: TWO MEMBER FRONT FIELD.
        Confidence in FVT setting is: MED
           [The pattern "FV-2" means that the finite
           verb must be the second element in the
           clause, but here it is preceeded by two
           elements, a definite syntax error.]
Due to: Possible English pattern in F.F. error
        Confidence in FVT setting is: LOW
           [Furthermore, the pattern looks
           suspiciously like English.]
Due to: Separable-prefix confusion
        Confidence in FVT setting is: NONE
           [As mentioned above, the prefix ought
           not to be separated from the infinitive
           "machen";  also, the prefix is in a
           random location, not even at the end
           of the clause.  At this point, the
           program gives up.]

  .. other comments:
           [A rehash of some of the important
           findings which might be of interest
           even if the play-by-play analysis is
           not consulted.]
DI position error, not separated from FV
Front field error (FV-2):  2 elements in F.F.

———— END OF RUN ————
```

In constructing the grader, I have been guided primarily by the need to approach the problem via the rules and structure of German grammar. The use of general linguistic information about German is indeed central to my work. The possibility of erroneous input seriously limits the applicability of an analysis based solely on general linguistic knowledge: the very grammatical constructions on which a general parser depends are subject to distortion due to just the kind of student mistakes that the program is designed to analyze. I attempt, therefore, to strike a compromise between generality and diagnostic capability. I supply a description of the expected sentence to aid the analytic routines, but have designed them to accept this description without being entirely bound to it letter by letter. Furthermore, the routines responsible for carrying out the various grammatical functions of the program are all constructed to deal with errors as a direct part of their functioning, not merely as a recovery measure.

The "correctness" of a student response in an instructional setting depends on a number of factors. Internal agreement rules are always considered, often subordinating the check on conformity with the expected answer. Partially correct answers are acknowledged as such; errors due to the propagation of misconceptions are so identified (rather than being counted as additional separate errors); plausible explanations are offered for many "incorrect" forms.

The program is very flexible in its ability to follow the student through many kinds of distortion and error. It is precise, in reporting independent errors independently. It is vigilant, insisting always on checking the most simple rules as well as the more complex ones. It is thorough, gathering information from one response on a large number of different aspects of student performance, through its ability to analyze an entire sentence. I believe that this grader will develop into an invaluable constituent of a computer teaching system which is truly able to adapt to the student.

I regret that this project must be so intimately involved with German. Natural language being the complex entity that it is, one cannot expect a simple theory to be at all sufficient. In the course of accommodating the special cases which together make up German grammar, I have evolved various general principles, which will be presented below. Some are specific to German, but many are of much broader applicability.

Chapter Two
LITERATURE REVIEW

This dissertation involves analysis of the German language, and thus belongs in a general way to the set of projects concerned with the analysis of natural language. In its development to date, my project has involved relatively simple grammatical constructions, placing few demands on the relatively modest parsing scheme I have employed. Certainly, modern work in linguistics has helped my project; in particular, the use of structural linguistics natural language grammar texts has been quite helpful. I have been able to use the grammatical schemata of the Stanford German text <ref. 10.1; Lohnes and Strothmann, 1968> extensively, with additional help from the Reference Grammar <ref. 9.1; Lederer, Schulz and Griesbach, 1969> (see Appendix A for more details). The suitability of these grammars, combined with the current simplicity of the grammatical constructions handled by my program, has greatly lessened the need to turn to more formal grammatical techniques.

In an important respect, my work stands apart from the other natural language analysis projects. Despite a great diversity among these projects, none manifest an interest in analyzing and grading incorrectly formed language.* Whereas I take the meaning of the sentence for granted, they are primarily interested in obtaining that meaning as the output of their analysis. Nor have I made much use of their theories of parsing, due mainly to the need to integrate error analysis procedures into every step of my program.

As this project continues to grow in grammatical complexity, I believe I shall be able to continue building on the foundations I have laid. As the grammatical complexity increases, some of the more sophisticated techniques developed by current researchers may well be of help in this project. In particular, the techniques used by Winograd <ref. 22.1; 1971> seem rather promising (though their response to erroneous forms would have to be adapted to my needs).

Education, particularly Computer-Assisted Instruction ("CAI"), is the field closest to this project. My interests are in grading, with eventual applications to teaching. Most CAI projects are primarily concerned with the operational aspects of teaching, and have given little attention to elaborate grading techniques. Still, there are a few comments I should like to make, concerning certain salient points:

---

* Not even those projects connected with Computer-Assisted Instruction.

— techniques of response recognition, particularly for large, structured answers;

— error analysis, providing interpretations for specific kinds of wrong answers.

I have consciously incorporated a substantial amount of knowledge of German grammar into the computer program. In so doing, I join a growing number of projects, chiefly in Artificial Intelligence, which believe that the computer must know what it is doing if it is to function intelligently.* I am convinced that Computer-Assisted Instruction can approach the level of a "good" human teacher only when the computer "knows what it is doing". We expect our human teachers to be trained in their subject, and look down on those who are one lesson ahead of their class. The computer will need at least as good credentials.

Traditional CAI projects generally do not attempt to have the computer "understand" the subject domain. These programs do, perforce, analyze answers, and sometimes draw conclusions from that analysis about what the student had in mind. A brief look may be of interest.

A large amount of CAI material is built around a fixed recognition scheme: the student responds in a multiple-choice mode, or in what I term "hidden

* See, especially, Winograd, who makes this point quite eloquently for understanding natural language. <ref. 22.2>

multiple-choice" in which his supposedly free response is matched against pre-specified alternatives. These schemes are attractive in allowing the instructor the freedom to associate arbitrarily complex interpretations to specific behavior.

For example, consider a simple exercise:

```
PRESENT TO STUDENT: "I see the book"
PRESENT TO STUDENT: "Ich sehe ____ Buch"
   REQUEST ANSWER
CORRECT ANSWER: "das"; (TYPE "correct", PROCEED)
INCORRECT ANSWER:  "die";
      TYPE "the noun 'Buch' is neuter, not feminine"
INCORRECT ANSWER:  "der";
      TYPE "the noun 'Buch' is neuter, not masculine"
```

The general failure to search out and use error interpretations may be due in part to the nonautomatic nature of the tool provided to the lesson programmer: each answer must be individually anticipated and accounted for. The construction of complete and accurate programs can be extremely difficult. Note that even the above simple example contains two errors: the possible form "den" has been omitted; and the message for "der" neglects to note that its use would imply more than a mere gender error, but also a mistake in case or else the use of an incorrect form for the masculine accusative article.

Another point worth noting is the extreme difficulty of applying simple recognition procedures to multi-element responses. When the multiple elements form a sentence in a natural language, the analysis is particularly demanding, due to considerations of both sequence and structure. Some valiant projects have come up with analysis techniques good enough to run experimental courses teaching natural languages. These techniques involved the introduction of at most the barest rudiments of subject-matter knowledge into lower-level analysis; mainly the programs have relied on subject-independent calculations, including spelling checks and sensitivity to permutation of the elements making up the response. These techniques lack reliability in a structured environment: they are not certain to reach the right conclusions about an answer, and they may be easily misled. Not surprisingly, they also are thin in diagnostic abilities.

One such project involved an experimental course in German, sponsored by IBM-Yorktown in the mid-1960's. <ref. 1.1> Their program was able to cope with multiple errors, though on a manual basis (each error had to be individually anticipated). For publication, they present the sentence "Er kommen heute in der Stadt" ("He is coming to the city today"), and observe that "two unrelated errors are present." Unfortunately:

> The first, a particular wrong inflection of the verb "kommen", was not forseen and looked for by the programmer; it was detected by the machine as essentially a spelling error...
> <ref. 1.2; Adams, 1968>

Their program also had a capacity for handling permutations in word order; but the algorithm takes no account of the structure of the German language, being concerned only with matching strings of characters and words.

The French project at IMSSS,* Stanford <ref. 14.1>, recently provided a beautiful example of the pitfalls accompanying a nonstructural permutation analysis. A student was expected to produce the sentence "Georges est un enfant" ("George is a child"), but apparently forgot that no article is required for a proper noun, and typed "Un Georges est un enfant" (literally, "A George is a child"). Because of the similarity of the two articles, The program was able to match the student's "Un Georges est" with the expected "Georges est un". After it determined that a permutation of order had taken place among these three words, the program went on to complain that the second "un" of the student's answer was an extra word! <ref. 14.2; Scholl, 1972>

Seeking greater flexibility, some researchers have used a generative approach to CAI. This technique involves

---

* Institute for Mathematical Studies in the Social Sciences

a program capable of generating exercises in some automatic fashion. These programs incorporate a rudimentary knowledge of the subject matter into the generation algorithm, as they must be able to compute the correct answer. (Note that it is not necessary to have the ability to solve the problem in order to generate it.) Rarely is the automatic portion extended to provide even the simple alternative-answer recognition capability of the fixed-recognition systems.

Siklossy presents a representative generational program, accompanied by a call for the general incorporation of subject-matter knowledge, in his article "Computer Tutors That Know What They Teach." <ref. 15; 1970> At first reading, his modest project appears to follow the same general lines I espouse. His program uses its knowledge of basic set theory (its subject-matter) to analyze answers for correctness, and also to dynamically explain errors. Siklossy's main concern, however, lay in being able to respond to questions posed by the student. Perhaps for this reason, his program as described stops short of trying to work with mistakes which are due to misconceptions rather than merely faulty technique. The program dynamically figures out errors of technique, but does not explore alternative explanations which might account for the observed response. Thus, for instance, consider a problem in which the student is asked to form the intersection of two sets, and produces instead their union. The program will happily use its symbol manipulating abilities to analyze the response, and produce an error report exhaustively listing all the elements of the student's answer which are members of one set but not the other. No test is made to see if the too-large answer constitutes a correct set union. Similarly, if the student commits a common error in forming a union, listing some members of the intersection twice, the program rejects his answer with a peremptory "not a set".

I would be quite optimistic about the prospects of expanding his program along the lines discussed in the body of this report, especially to include attention to the possible existence of conceptual errors and in general to attempt analysis of errors for probable cause. His case for the value of incorporating subject-matter material is well taken, and has resulted in a sound program on which one could base my kind of response analysis.

Two large projects, by Wexler and Carbonell, come reasonably close to my general analysis goal, albeit in a totally different subject area. <refs. 21.1 & 2.1; 1970> These two are basically Artificial Intelligence programs. They incorporate a large, complex network of data from the subject matter, and routines to manipulate it to both generate problems and analyze answers. The ability to extract from the data-base an alternate explanation for a student's "wrong" answer is just the kind of performance I am after. To form an informal example from Wexler <ref. 21.2>, consider that the student is to be asked about the capital of Massachusetts. His response would have to satisfy criteria such as:

1) must be a city
2) must be in Massachusetts
3) must be the state capital

If the student responds "Springfield", the computer will find:

a) Criteria (1) and (2) are satisfied (Springfield is a city, in Massachusetts), but (3) fails, it is not the state capital.

The computer might also find:

b) Criteria (1) and (3) hold while (2) fails, as a city called Springfield is the state capital of Illinois.

One can see how appropriate messages can be constructed from the information discovered by the two analyses. The format of the information would facilitate the recording of general trends, should the instructional program so desire. For instance, (a) "student does not know which city in a state is its capital," or (b) "student tends to confuse the capitals of different states." Both Wexler and Carbonell have been interested in error analysis primarily to enable their programs to reply to the student with more interesting comments, and so to help him with his immediate difficulty. In particular, they do not address a question which is of major importance in my work: how to deal with the ambiguities resulting from multiple interpretations of responses, as in (a) and (b) above.

I should add that Carbonell does devote several pages to discussing error analysis, in spite of a strong statement of policy which has often accompanied a profound disinterest in considering student errors:*

> If problems develop the ultimate objective is for the student to overcome them, not for the teacher to diagnose them. <ref. 2.2>

Carbonell clearly recognizes the existence of different sorts of errors:

> In the case of symbolic answers, if a student asked about the capital of Argentina responds Brazilia, he is not making as serious a mistake as that made if he would answer Brazil (which is a country).
> <ref. 2.3>

He suggests that a classification ("taxonomy",<ref. 2.4>) of errors is needed to provide a good theoretical basis for diagnosis. This is presented largely as suggestions for future work, albeit with a good dose of optimism:

> ...if human teachers have serious difficulties in dealing with causes of errors, it may be premature to expect a highly sophisticated behavior of CAI programs in this respect. On the other hand, ISO

---

* See a similar quotation from a language teacher, Chapter 6.

> [Information-Structure Oriented] programs, of which SCHOLAR is the first example, will provide a rich environment for research on error types, detection, and diagnosis, and on consequent teaching strategies.
> <ref. 2.5>

I agree, and present my project as just that kind of research.*

Concerning meaningful interpretation of errors, a paper by Suppes and Morningstar <ref. 17; 1970> suggests research along a line similar to one I have followed. They postulate a cerebral automaton to account for each facet of the student's performance. In simple arithmetic, for example, one automaton might perform the addition of single digits, while another would take care of the carries in multiple-digit addition problems. Following this theory, incorrect performance will occur in two ways:

1. There may be a specific malfunction in an automaton, such as a failure to process carries correctly.
2. The student may select a correctly functioning automaton which happens to be inappropriate, such as one which subtracts instead of adding.

In either case, the teacher faces a well-defined task in helping the student with his misconception.

* I understand that Carbonell has delved further into the question of error analysis in his more recent work. <ref. 3; private communication, 1972>

In its general statement, this theory corresponds rather well with what I am doing: we both want to find formulations of what has happened which can account for the observed deviant performance. Unfortunately, Suppes does not deal with the problem of how to go about discovering the existence and characteristics of these automata. Nor does he discuss the problem of reconciling alternative explanations for specific performance.

Looking to those engaged in language teaching, particularly German, I find some encouragement, a little help, and much challenge. An experienced teacher can easily grade a student's answer, though without being aware of exactly how he goes about doing so. Perhaps an intensive study of the way in which humans analyze erroneous sentences would produce some information relevant to my project; I am not optimistic, at least not until the theoretical work on natural language has progressed much further.

Human teachers do offer explanations for wrong answers, again at an intuitive level. In Chapter 6 I will enter into a detailed discussion of this behavior and its relation to the current project. At that time, I will also review some recent linguistic work in the same vein as Suppes' proposal.

PART TWO

Matching the Student's Sentence with the Expected Words and Structures

Chapter Three
BASIC METHOD

The grading program which I have constructed represents a compromise between the wide-ranging capabilites of the human teacher, and the limited scope of the simple teaching machine. This program contains a large quantity of information about German grammar, which I augment with information about the specific sentence I expect to be analyzed. The resultant system is powerful enough to accept a wide range of variation in the student's response and still produce a meaningful diagnostic report.

The information given to the program is not the exact "correct" sentence, but rather a description of it. This description consists primarily of the major words and structure of the expected sentence. The program's strategy is to match the key elements of the description to what actually appears in the student's response. Each instance of a successful match provides a foothold from which the program can work to apply its general rules to analyze and report.

In the course of setting up the strategy for the grading program, I have perforce made certain assumptions about the nature of the problem. These assumptions also provide a convenient framework for discussion.

My basic premise is that the student will produce a specific sentence. Put differently, the student's performance is sufficiently constrained as to warrant assumptions about the specific vocabulary and structure he will use in forming his sentence. These particular assumptions carry through right into the heart of the program's operation: when a word in the student response can be identified as one of those whose presence was expected, the program can immediately orient itself with the structure of the sentence simply by referring to what it knows about the expected structure surrounding the word in question. (The assumptions of vocabulary and structure take concrete form in the specification of the expected sentence, which is supplied to the program prior to the analysis.)

The immediately following discussion will delve further into the operational characteristics and features of this identification, and consider the type of information which can be obtained using a structural orientation. In the next chapter, I will discuss the limitation imposed on the student by my basic premise.

I do not assume that the student will correctly follow rules of inflection; on the contrary, the program design places as little reliance as possible on inflection for guiding the analysis. The matching algorithms must, and do, allow for variation in form due to inflectional error by the student: essentially noise in the matching process.

For lexical words (nouns, verbs, adjectives) the noise is usually not too great, and the word can still be recognized reliably. A more detailed description of the matching algorithms follows later in this section. I would note here, in passing, that the matching process is somewhat facilitated by the restriction to specific vocabulary.

Function words (articles, prepositions) pose a different problem for the recognizer. The program is able, of course, to determine that a particular word belongs to a given class of function words. But since these words owe almost their entire form to inflectional dictates, reliable identification must be based on more than orthographic form. Furthermore, function words (as their name implies) do not usually convey meaning by themselves, but by their association with lexical words and phrases. I rely heavily on structural information in locating function words, making particular use of their predictably close association with lexical words which can be differentiated by form and meaning.

I have made a basic structural assumption, that the student will keep the clause-elements of the sentence mainly intact; that he will place their constituent words in a predictable order. A noun phrase, for example, must appear with the article preceeding the noun, or maybe even missing, but never following the noun. This assumption provides the power needed to cope with function words.

As the program proceeds through the sentence, it is primarily looking for lexical words, which it can recognize directly. By referring back to the descriptive information, the program can tell what structure surrounds the word it has found (also as outlined previously). If this structure includes function words, the program need only look for them in the place where they should occur. Any candidate in that location will be accepted if it is the right kind of word. There may be a secondary check to see that it does not differ too widely from the expected word, but that check will be analytic, not orthographic; i.e., based on the infectional characteristics of the word, not on its spelling. Thus, for example:*

Expect: "das Buch .." ("The book ...")

Actual: "die Bucher .. "

Examine: "die"; Conclude: article;
no identification yet

Examine: "Bucher"; Conclude: matches noun "Buch";
Noun phrase was "das Buch", so look
backwards for article preceeding
the noun.

Reexamine: "die"; Article, claim for noun phrase

The student may supply some extra words, in spite of my basic premise about close specification; or he may omit some words, particularly function words. The program must

---

* see also example given in Chapter 5, of actual program trace from just this kind of situation.

of course report on the occurrence of any such deviation from the expected. Internally, there is the difficulty that the program might push its search for a function word too far and claim a word that really belongs to another clause element. For the present, I solve this problem with another assumption, that a clause element must be a contiguous block.* Thus if a word belonging to some other element is encountered during a function word search, that search must fail. (A clause element can, of course, be expanded below by the inclusion of dependent clauses as modifiers; the program is able to accomodate itself to this situation.)

The order of elements themselves in the clause, on the other hand, need not meet any assumption imposed by the analyzer. Here the issue is even more critical than for inflection. I continue to assume that any rule will at some time be violated by a student. The German language, moreover, has many extremely flexible rules governing the placement of clause elements within the sentence. Even assuming that the student's performance will be correct would not allow predictions about element ordering. I have had the program explicitly avoid element-order assumptions; the scope of any simple structural inferences gained by the recognition of an expected word is limited to the clause element of which that word is a member.

* This assumption may be a bit too strong in some instances. I have one example in which a student wrote "Wieviel willst du Geld?" ("How much would you like money?"), thereby splitting the clause element "wieviel Geld" ("how much money").

Pronouns represent a slight embarrassment in the scheme I have employed. Their form is quite sensitive to inflectional alteration, like function words; but pronouns stand alone in a clause element (sometimes with the help of a preposition), and so there is no structural context to aid the search. The program can generally recognize look that a word is a pronoun; the problem is making a correct identification, particularly when the expected sentence contains more than one pronoun. I am forced to assume that the inflection, if not exactly what was expected, will nonetheless be "close." The program, accordingly, is equipped to decide which of two possible choices is "closer."* That heuristic seems to be doing a good job.

Word Recognition

This section considers the problems of word recognition in greater detail, with particular attention to the effects of inflection. The analytic question, of how inflectional forms are checked for correctness, will be the subject of a later section. (Chapter 7)

The recognition problem is eased by the practice of supplying the program with the vocabulary from which the student sentence should be formed. Since only that vocabulary is considered, the scope of the search is kept

---

* See Chapter 7.

quite small, with immediate benefits in the amount of time which can be devoted to each comparison. Before doing any scanning of the student's sentence, the program makes up specialized search routines for each expected major word, taking maximum advantage of the information it has on each word. In particular, a different strategy is employed for each word type, to be best able to accomodate inflectional noise. These routines do not try to take advantage of the limited size of the vocabulary list to make less rigorous comparisons; the program will properly detect an unexpected word as one it cannot recognize.

At this stage of development, the program incorporates a simplifying restriction on the kind of sentence it can analyze, a restriction which will certainly need to be relaxed in the future. Each lexical word in the expected sentence can occur only once. Sentences like "The green car is faster than the red car" cannot presently be handled. The benefits for the analyser are nontrivial. In forming a firm identification of a word in the student sentence with a word of the expected sentence, inflectional changes must be taken into account; but nothing beyond the specific word need be considered. In particular, the program need not worry about the following or preceeding word.

Misspelling will upset the program as it is currently constituted. This capability was omitted thru a decision on allocating my resources; it is less innovative, and not crucial for German (whose spelling is largely phonetic, as compared to English). See also Chapter 10 for some suggestions on future work on this problem.

Inflectional changes, both correct and incorrect, constitute the major complication in identifying the words in the student sentence. Happily, many major words inflect, if at all, by suppletion, i.e., by adding an ending to an invariant stem. Such words can be located by merely ignoring the ending (though saving it for later checking, of course). For those words which show inflectional changes in the stem as well, the change is usually fairly small and more or less predictable. The few pathologic words must just be handled as special cases.

Some examples:

Simple suppletion in noun plurals:

das Kind, die Kinder (child, children)

der Knabe, die Knaben (young boy)

Stem change in noun plurals:

der Mann, die Maenner (man, men)

Greater change in noun plurals:

der Schutzmann, die Schutzleute (police)

I now proceed to a detailed description of the word matching algorithm for each part of speech.

Lexical Words:

Adjectives: Adverbs and predicate adjectives are completely uninflected in German. The matching algorithm for these words consists of a full word equality check against the expected text.

Attributive adjectives inflect by pure suppletion. The program does not yet handle this category, due to the complications of interpreting inflectional errors. The match will be a stem-plus-ending comparison: a simple equality match of the expected text against the actual word (stem) allowing for an arbitrary ending on the latter.

Nouns: Nouns inflect, usually, for the plural, and that generally by suppletion; the stem vowel is umlauted at the same time in roughly half of the nouns. My match allows for an arbitrary ending, as the student may well not remember how to form the particular plural; similarly, I have a flexible check for stem vowel, allowing almost any vowel or vowels. The program could perhaps check only for the correct vowel with or without an umlaut, but does not make that restriction at this time. The flexibility in stem and ending is extended even when searching for singular nouns. The student might, for some reason, supply a plural instead; also the student might not remember the correct singular form of the stem, especially if he had recently worked extensively with the plural.

Specifically, the match pattern consists of four fields:

1. (front part of word) characters specified exactly;
2. (stem vowels) match any one or more vowels;
3. (rear part of word) characters specified exactly;
4. (ending) match anything.

For example, "der Mann" (man), plural "die Maenner"

Pattern is: "M" <STEM VOWELS> "NN" <ENDING>

Match on "Manner" with <STEM VOWELS> = "A"
<ENDING> = "ER"

If a field does not exist, it is specified as blank, as for "der Artz" ("doctor") in which nothing preceeds the stem vowel "a".

Verbs: Verbs show some of the most complicated inflectional changes of all German words. I have allowed, so far, only for infinitive and present tense forms; though it is not the recognition problems but the error analysis which has delayed expansion into the other tenses. Verbs have endings; if irregular, they also must change their stem vowels and occasionally even a few consonants. As an additional complication, there may be one or more prefixes: separable prefix adverbs, the preposition "zu", (and for past participles, the form "ge"). As with nouns, I have allowed for incorrect variations on any inflectional pattern: any vowel substituted in the stem, arbitrary prefix and ending, even some stem consonant changes.

My experience so far has not encountered any false matches. For some very irregular verbs, the reliability of the algorithm may need some bolstering through the addition of stronger conditions. For instance, the verb "essen" (to eat) has only a double-S invariant; the present program will match any word with a vowel followed by "SS". To increase the reliability of the match, a plausibility check should be made on the prefix and ending. Some restriction on vowels may also be helpful; in the example, "u" or "ue" would seem unlikely candidates for "essen".

A special matching algorithm will be needed for the very irregular auxilliary "sein" (to be), whose forms show no resemblance at all to the infinitive.

Specifically, the match pattern is set up with five fields:

1. (prefix) matches anything zero or more letters;
2. (front) exact character match on front part of stem;
3. (stem vowel) one or more vowels;
4. (stem consonant) exact match on consonants following stem vowels; more than one set may be given for verbs that may change this part of stem;
5. (ending) matched anything left over.

For example, "kaufen" ("to buy"):

Pattern is:

<PREFIX> "K" <STEM VOWELS> <STEM CONS. "F"> <ENDING>

Match on "kauft" with <PREFIX> = nil
<STEM VOWELS> = "AU"
<STEM CONS.> = "F"
<ENDING> = "T"

For example, "nehmen" ("to take"), an irregular verb:

Pattern is:

<PREFIX> "N" <STEM VOWELS> <STEM CONS. "HM","MM"> <ENDING>

Match on "abnimmen" with <PREFIX> = "AB"
<STEM VOWELS> = "I"
<STEM CONS.> = "MM"
<ENDING> = "EN"

Function Words:

The recognition problem for function words is somewhat different from that for lexical words. The main routines of the program use structural information, rather than purely expected form, in locating the function words. The recognition routines are called on to determine if a given word is a preposition (or article), and, if so, to find out the word's characteristics.

Prepositions: For prepositions, the recognition problem is quite simple, as there is no inflectional change. What the student might well do instead is to substitute a different preposition. In any event, simply checking a list of all prepositions suffices nicely.

Contraction of prepositions with articles presents an annoying complication. I check suspected words against a short list of legal contractions. If one is found, it is a simple matter to break it into the two constituents and analyze them as if separate.* Errors in forming contractions are nearly in the realm of spelling mistakes, and are not yet handled; Nor are contractions of prepositions with pronouns.

Articles: Articles are very highly inflected words with a plethora of orthographic forms. My recognition algorithms allow for any inflectional form of any kind of article. (The major problem, of interpreting what the student has done, rests on the inflection-checking logic.) For a definite article to be recognized, its form must correspond exactly with one of the actual definite articles, as "d-" is not a sufficient stem for generalized checking; for other types, the recognition is based on stem plus arbitrary ending (just as for attributive adjectives).

Algorithm: first search the list of definite articles for an exact match; if so, TYPE is "definite", ENDING is taken from the list..."die - e", "das - es", "der - er", "den - en", "dem - em". If the word is not a definite article, then try for a stem-plus-arbitrary-ending

* this check is actually made on every word, before any other checks are made on it. The inefficiency of the extra search are made up for by avoiding problems of crossed pointers when splitting a contraction.

match using all the other article stems; set TYPE to "EIN-WORD", "DIES-WORD", or "POSSESSIVE" accordingly.

Pronouns:

Pronouns present a major problem to the program as a whole, but not too much headache for the recognizer, which is only asked whether a particular word is a pronoun. German pronouns inflect mostly by drastic changes in form, so a simple check against a list of all personal pronouns is sufficient to make the rough identification. A slight complication occurs in definite pronouns, whose forms are mostly the same as for the definite articles. Again, the question of is it or isn't it can be resolved by checking against a list of all legal possibilities; the stickier question, as to whether the word is a pronoun or really just an article, is taken care of by the pronoun search logic. (See Chapters 5 and 7 for more details.)

Chapter Four

IMPLICATIONS OF THE LIMITING ASSUMPTIONS

In the previous chapter, I set forth the various assumptions under which my program operates. These assumptions are acceptable as defining and delimiting a specific project; however, the relation of this project to the outside world requires some further discussion. In particular, I should like to discuss the definition of the boundary created by the operating assumptions, and consider the justification of these assumptions from an educational point of view.

Boundary:

The program at this time adheres rather closely to the assumption that the student is asked for a specific sentence and will endeavor to produce that sentence. The program conducts its search and analysis under close guidance from the expected answer. Although it is sensitive to many variations, the program does not attempt any kind of general analysis which would encompass an arbitrary German sentence. In particular, there is little attention paid to determining the meaning of the student's sentence as possibly different from the expectation. There is a basic problem at issue, which has to be solved by the basic philosophy of the grading project: how to distinguish distortion [grammatical error] from intentional semantic change. I believe that some headway can be made in making such a distinction, but for the present project have made the program concentrate only on error analysis; in fact, I have designed it to operate in spite of possible alternate meanings and grammatical ambiguities which might otherwise have frustrated the analysis.

Currently, any deviations from the expected formulation, either in individual words or in structure, are analysed as local errors only. Certain manglings can, of course, result in a sentence with a different meaning, possibly expressed in "correct German" if taken alone. For example:*

-sentence-	-program will report-
1. Die Frau sieht ihn. ("The woman sees him.")	Correct
2. Die Frau sieht ihm.	Pronoun wrong, perhaps dative instead of accusative.
3. Die Frau sieht er.	Pronoun wrong, perhaps nominative instead of accusative.

Sentence 3 happens to be also a correct way of expressing the thought "he sees the woman," since "die Frau" can be either nominative (subject) or accusative (object). The program will not be distracted by this different meaning.

* for conciseness, the program output is paraphrased. Actual examples of the diagnostic comments can be seen elsewhere.

It need not remain totally ignorance of the alternative, either. In future work, I would like to explore incorporating certain special checks depending on specific errors. Various heuristic approaches will be needed; the above example could have been caught by checking possible inversion whenever an object appeared in nominative case.

Limitation: theoretic

The assumption of a specific sentence, and the information provided to the program, provide a great deal of assistence to the analyser, but at the same time greatly restrict its generality. I have tended to justify the restriction as necessary to be able to cope effectively with a natural language situation, and indeed I am willing to make sacrifices in order to obtain powerful grading facilities. I believe there is a basic theoretical question here, as to whether extra information is really needed to recognize improperly formed sentences. A few comments may serve to illustrate the problem, and, with it, my choice of assumptions.

(An informal information-theoretic argument)

As allowance is made for the possibility that the input will not conform to the rules of German grammar, supplementary information is needed to compensate for the assumptions we are no longer entitled to make. If inflections or word-order are possibly wrong, some other source must be found for the information that they provide about the meaning of the sentence. Turned around, the argument is that by having other sources of information, the grader can regard these rules as items to be checked for correctness rather than as crucial aids in completing the analysis.

A human teacher knows what he asked, and thus has extra, sentence-specific information available when he is grading the student response.

Limitation: education

It remains to examine the restrictions from an educational point of view. Is there any place for a grading program with such restricted scope that it can only handle one specific sentence at a time? Furthermore, what do language teachers feel about such restrictions? Both questions can be answered favorably: even though the project has been kept carefully circumscribed, it still has possible uses.

One common technique used by a modern approach to language teaching is to have the student "overlearn" basic sentences. These basic sentences, or parts of them, are used as "building blocks" as the student goes on to new material, producing a situation in which vocabulary will be naturally quite restricted. From one such text:

> This lesson is devoted to review ... of 85 sentences, of which some are identical with the basic sentences of the preceding

> lessons, some are new combinations of the elements of the basic sentences, and some contain material to increase your reading vocabulary. These sentences are in German in the left-hand column, with equivalent English sentences in the right-hand column. Study these by first reading the German sentences, referring to the English sentences only when you are uncertain. Repeat this study until you can recognize and understand accurately every one of the German sentences.
>
> The next step is to look at the English equivalent sentences and check your ability to produce the proper German sentence.
>
> <ref. 12.1; Rehder & Twadell, 1958>

In another text, the directions to the student for a particular exercise leave very little room for inventive work:

> Express the following sentences in German. This is not meant to be a translation exercise; the English sentences should "trigger" their German equivalents.
>
> <ref. 10.2; Lohnes & Strothmann, 1968>

A third example seems almost tailored exactly to fit the capabilities of my program. (The citation is from the introduction, a general description some of the material offered by the text, a beginning reader):

> ...synthetic exercises, made up of "dehydrated" sentences... The student is given the thought content and vocabulary of a particular sentence and is asked to reconstruct the sentence by "adding grammar."
>
> <ref. 19; 1967>

Philosophically, language teachers recognize that freedom of expression has its place. That place may not be everywhere:

> In most cases, putting the meaning across is a minor part of teaching a dialog.
> <ref. 8.1; Lado, 1964>

> We conclude that it is not the teacher's task in first-year German to chat discursively with his students, but to guide them in the controlled formation of habits. So far as the students are concerned, the material they are practicing <u>is the German language</u> for the time being. We need and probably should not go beyond this material; "other ways of saying the same thing" are a distraction, not an enrichment, for the beginning student.
> <ref. 12.2; Rehder & Twaddell, 1958>

I might add that I do not necessarily agree with all the opinions which I have cited, and would certainly be happy to have greater flexibility available. At the same time, I do not lose sight of some rather significant possible benefits possible even with the limitations intact. By being able to accept full sentences and monitor, automatically, a large number of different rules, the grading program immediately permits a more flexible curriculum than would be possible with a less versatile grading facility. I feel that I have already come far enough that research into using the grader in new flexible curricula can begin in earnest.

Chapter Five

PROGRAM ARCHITECTURE

This section describes the basic architecture of the program, and includes a fairly detailed discussion of the algorithm involved.

The program is given two major inputs: a description of the expected sentence, prepared by the teacher or helper; and the actual student sentence which is to be graded. The analysis takes place in four phases:

1. pass through the description of the expected sentence, to build a guiding structure;
2. main pass through the student sentence, recognizing key words and collecting associated words by directed search;
3. second scan of the sentence to collect pronouns and separable prefixes;
4. word-order analysis of the clause-element structure to which the sentence has been reduced.

Here is an example of the program's actual operation which I will use at several points of this discussion. The student is asked for "Er kauft einen Hut" ("He buys a hat"), and responds correctly. The following output represents just the grading comments; later in the discussion, I shall present the same analysis with the program printing internal tracing commentary as well.

```
———————— RUN ⁴1 ————————

Description generated by human expert:
(STMT (SJ ER) (VERB KAUFEN) (OJ ACC EINEN HUT))

SENTENCE INPUT BY STUDENT  ..
EINEN HUT KAUFT ER

NOW PROCEEDING WITH ANALYSIS

..........  SUMMARY REPORT FOR NOUN PHRASE INFLECTION
                ON EINEN HUT
          Best we can do guessing intent is: (ACC S M)
             CONFIDENCE LEVEL: VERYHIGH

..........  GENDER REPORT FOR NOUN: HUT ::
          STRONG CORRECT = M

..........  CASE REPORT FOR OBJECT: EINEN HUT ::
          STRONG CORRECT = ACC

..........  CASE REPORT FOR SUBJECT: ER ::
          STRONG CORRECT = NOM

..........  VERB ENDING CHECK .. OK!

..........  Phrase division done, sentence type report:
          Final verdict is FV-2 , level of confidence:
 VERYHIGH

          Play by Play account of analysis:
          Due to: INITIAL HYPOTHESIS , FVT set to FV-2
                  Confidence in FVT setting is: VERYHIGH

..........  Summary Comments
          Clause type correct (FV-2)

 ———————— END OF RUN ————————
```

Operation of Phase One:

The program begins with a pass through the description in order to build the complex structure it needs for attacking the student sentence.

For each clause of the expected sentence, a list of the important words* is collected into a list called WDLIST. At the same time, the program builds a structure for each clause element, containing the consitutent words and information which will be needed in the analysis. The structural details communicated in the sentence description play an important role in this stage of building. Cross-reference pointers are set up between the word entries in WDLIST and the corresponding clause element structures, and the latter are collected together into a SDLIST. Pronouns, because of their special problems, are collected not in WDLIST but in another place of their own, PNLIST.

At the end of Phase One, the sentence description consists of the accumulated WDLIST, SDLIST and PNLIST's for each clause. Specific information on the particular words, needed in various stages of the analysis, is available to the program by this time; it has either been given as part of the description, saved from previous work, or requested from the experimenter during Phase One.

The other main input to the program, the student sentence, is still but a simple collection of words. They are arranged in such a way as to facilitate the analysis.

To continue the previous example,
"Er kauft einen Hut." ("He buys a hat.")

---

* See Chapter 3 for definition of this concept

The initial description, as prepared by a teacher and given to the program, contains both the text and structure of the expected sentence:

```
(STMT (VERB KAUFEN)
      (SUBJECT
               (PRONOUN ER))
      (OBJECT ACC
               (NP
                  (ARTICLE EINEN)
                  (NOUN HUT)   )))
```

(The abbreviated descriptors "SJ" & "OJ" which appear in the full examples actually expand into the more explicit structure shown here. I have not placed much emphasis on making the program smarter about the input description, concentrating my efforts instead on the analysis of the student sentence given whatever seemed the most helpful description.)

Here is a complete listing of the WDLIST as set up for the analysis of "Er kauft einen Hut." It contains two search functions, NOUN-WD for the noun "Hut" and VERB-WD to look for a form of the verb "kaufen". The other information represents the parameters needed for the identification, particularly the patterns for guiding the match algorithm. The cross-reference pointers to the SDLIST terms are held in WS-7 (for the noun) and WD-6 (for the verb).

```
((NOUN-WD WD-11 HUT ((H) ((SV . VV) T (END))) (WS-7))
 (VERB-WD (QUOTE WD-5)
          (QUOTE KAUFEN)
          (QUOTE ((PRE) K (SV . V) (SC (F)) (END)))
          (QUOTE WD-6)))
```

Here are the SDLIST terms, again as functions with a large number of parameters. The COMELTA-SD functions handle either subject or object —— the sixth argument tells which. The last argument is itself a structure of functions, describing the constituent words or phrases. The first argument of each of the top-level SDLIST functions is the name of the variable holding the cross-reference pointer from the key WDLIST or PNLIST function. (My comments in []'s).

```
(
[WS-2 points here:]
 (COMELTA-SD (QUOTE WS-2)
             (QUOTE PP-3)
             (QUOTE NOM)
             (QUOTE (SUBJECT SPEC))
             (QUOTE NIL)
             (QUOTE SUBJECT)
             (QUOTE (
                     (PRONOUN-SD WD-4 ER (3 NOM S M PER)))))
[WS-7 points here:]
 (COMELTA-SD (QUOTE WS-7)
             (QUOTE PP-8)
             (QUOTE ACC)
             (QUOTE (OBJECT SPEC))
             (QUOTE S)
             (QUOTE OBJECT)
             (QUOTE (
                     (NP-SD SD-9
                            (ARTICLE-SD WD-10 EINEN)
                            (NOUN-SD WD-11 HUT ((CSSV . U)
                                (XGENDER . M) (HNOUNFLAG . T)
                                (XNCLASS . O))))))))
[WD-6 points here:]
 (VERB-SD (QUOTE WD-5)
          (QUOTE KAUFEN)
          (QUOTE VERB1)
          (QUOTE CL&VERB1)
          (QUOTE  ((SS (F F SC) (AU AU SV)) (CLASS . 1)))))
```

The PNLIST looks just like the WDLIST, except that it has only pronoun terms at the start. The value of WS-2 is a pointer to the SDLIST term.

```
((PRONOUN-WD WD-4 ER NIL (WS-2) (3 NOM S M PER)))
(PP-3 PP-8)
FV-2))
```

Initial form of student input:

```
((WORD ICH)
 (WORD KAUFE)
 (WORD EINEN)
 (WORD MANTEL))
```

(Actually, the "exploded" form of each word is also included, to boost operating efficiency; thus,
(WORD ICH (I C H)).)

Each element on the WDLIST is concerned with one of the major words of the expected sentence. The element specifies two things, a search function and a cross-reference pointer. During the second phase of the program, the search routine will be given a sequence of characters: a word from the student's sentence. The routine must decide if that sequence matches its target word, i.e., the particular major word from the expected sentence with which it is associated. This identification task is entrusted to a matching algorithm specially tailored to the particular word-type in question. (see Chapter 3, above, for a description of these algorithms). [The name of the particular function is usually that of the word-type, suffixed with an identifying "-WD", as

"NOUN-WD".] The necessary information specifying the specific word to be found is all assembled by the Phase One processing.

The cross-reference pointer will be discussed more fully below.

Phase Two:

After the first phase has finished, Phase Two begins by activating the clause executive function with the WDLIST, SDLIST AND PNLIST for the main clause. The basic scan of the sentence is undertaken at this time, proceeding from left to right. Pursuant to the strategy discussed above, the top level activity is concerned with finding major words. The major word recognizers are collected on the WDLIST; the job of presenting the next word of the sentence and deciding what to do with it falls to a function named RUNWD.

RUNWD:

1. No words left: set FINISHED flag for clause executive; exit.

2. See if word is a function word (article, preposition). If so, remember it, go on to next word.

3. For each element of the WDLIST, activate the match function. If one likes the current word:
   3a. remove that entry from WDLIST;
   3b. set up pointer into corresponding SDLIST entry (variable is "WD-SD PTR"); exit.

4. Test if word is a pronoun: note, go on to next word.

(step 5 is relevant only for relative clauses...)

5. No match yet...look through all other WDLIST's, as in step 3. if one matches, then:
   5a. set NEWCLAUSE flag for clause executive, saving the name of the clause containing the lucky WdLIST element;
   5b. set FINISHED flag if moving up a level;
   5c. do steps 3a and 3b above;
   exit.

6. If none of the above, complain about extra word.

The recognition functions on WDLIST do very little analysis of the words they find. Their importance lies in their links to the analysis structures on the SDLIST. Each word (on WDLIST) has a pointer to the structure for the clause element of which it is a part. This pointer leads to the head of the clause element of which the word is a part, not to the SDLIST function for the word itself. In the case of a clause element with more than one major word, all corresponding WDLIST entries would have the same pointer. A clause element's SDLIST entry reflects the structure of the element; entering it from the top gives the program full analytic control of the context while it is processing any part of the structure.

Here is the previous example, this time including output relevant to the internal workings of the program. Some of the comments refer to features not yet discussed. For the present, I would call attention to lines beginning with one of the following legends:

"DOWD just did " — identifies a WDLIST function which successfully matched a word of the student sentence;

"SDDO: " — beginning execution of an SDLIST structure which was activated by a cross-reference pointer;

"THDO: " — beginning execution of individual functions within the SDLIST entry currently active.

——————— RUN 1 ———————

```
New tasks: ((V-FVENDCK))
DOWD just did NOUN-WD

SDDO:  COMELTA-SD (WS-7)
THDO: NP-SD
THDO: ARTICLE-SD
THDO: NOUN-SD
THDO: ARTICLE-SD
New tasks: ((NICK TEM-12 TEM-13))

..........  SUMMARY REPORT FOR NOUN PHRASE INFLECTION
                  ON EINEN HUT
             Best we can do guessing intent is: (ACC S M)
                CONFIDENCE LEVEL: VERYHIGH

..........  GENDER REPORT FOR NOUN: HUT ::
             STRONG CORRECT = M
Task just done: NICK

..........  CASE REPORT FOR OBJECT: EINEN HUT ::
             STRONG CORRECT = ACC
DOWD just did VERB-WD

SDDO:  VERB-SD (WD-6)
STEMTYPE: *
Stem-ending char ((2 P 7) (3 S 7))
New tasks: ((VERB1))
Task just done: VERB1
THDO: PRONOUN-WD
End of search for pronoun ER level = 4 GOOD
THDO: PRONOUN-SD

..........  CASE REPORT FOR SUBJECT: ER ::
             STRONG CORRECT = NOM
New tasks: ((POSCK-SUBJ))
Task just done: POSCK-SUBJ
JSCHR = (* * 7)
JECHR = (3 S 37)
JSECHR = (3 S 7)
```

```
..........  VERB ENDING CHECK .. OK!
Task just done: V-FVENDCK

..........  Phrase division done, sentence type report:
        Final verdict is FV-2 , level of confidence: VERYHIGH

        Play by Play account of analysis:
        Due to: INITIAL HYPOTHESIS , FVT set to FV-2
                Confidence in FVT setting is: VERYHIGH

..........  Summary Comments
        Clause type correct (FV-2)

 ---------- END OF RUN ----------
```

To follow through an SDLIST function in operation, consider what happens when the noun "Hut" is found. The program trace at that point contains the message "DOWD just did NOUN-WD". That function returned a cross-reference pointer, namely WD-7, which the program begins to follow ("SDDO: COMELTA-SD (WS-7)" message). The full structure of that element is given above; here is an abbreviated form for quick reference:

```
COMELTA-SD OBJECT ACC
     NP-SD
          ARTICLE EINEN
          NOUN HUT
```

Following the execution, with actual program trace shown as comments in square brackets:

[SDDO: COMELTA-SD (WS-7)]

COMELT-SD is a function, which, when entered, sets up the program and grammar context for the analysis of the clause element...the name, "object", expected case, "acc", and other variables to hold the results.

[THDO: NP-SD]

NP-SD is entered, and like COMELT-SD, sets up a program and grammar context for the analysis of a noun phrase. It then defers further action while the phrase constituents are handled.

[THDO: ARTICLE-SD]

ARTICLE-SD is entered. This SDLIST structure was activated because the noun "Hut" was found. Until the noun's analysis is accomplished, this routine graciously defers.

[THDO: NOUN-SD]

NOUN-SD is entered, and completes the analysis of the noun "Hut", determining its inflectional characteristics for further checking later.

[THDO: ARTICLE-SD]

ARTICLE-SD is tried again, is now happy to do its job. It scans back to pick up the article "einen" immediately before the noun, analyzes its inflection, and returns.

[New tasks: ((NICK TEM-12 TEM-13))]

NP-SD now resumes, checking the inflectional status of the entire phrase using information returned by ARTICLE-SD and NOUN-SD.

```
[..........  SUMMARY REPORT FOR NOUN PHRASE INFLECTION ]
[                  ON EINEN HUT                        ]
[           Best we can do guessing intent is: (ACC S M) ]
[              CONFIDENCE LEVEL: VERYHIGH              ]
[                                                      ]
[..........  GENDER REPORT FOR NOUN: HUT ::            ]
[           STRONG CORRECT = M                         ]
[Task just done: NICK                                  ]
```

With NP-SD finished, COMELT-SD can do what analysis it has for the whole element, before it, too, returns.

```
[..........  CASE REPORT FOR OBJECT: EINEN HUT ::   ]
[           STRONG CORRECT = ACC                    ]
```

Data structure:

The results of the analysis are saved in two ways. Specific grading observations, such as would be of most interest to a teacher, are expressed as (English) messages typed immediately at the terminal. All during the analysis, the program internally adds the information it finds to a data structure representing the student's sentence. This structure is represented by a list, which at the beginning consists of only the individual words input by the student, as given above. The search functions (from WDLIST) will change the particular word entry to reflect what they have discovered and decided about the word. Thus the simple

```
(WORD (KAUFEN (K A U F T)))
```

becomes:

```
(VERB (WORD KAUFT (K A U F T))
      ((SC F) (SV A U)
              (END T)
              (CLASS . 1)
              (END . T)
              (ENDCHR (3 S 37) (2 P 77))
              (SECHR (2 P 7) (3 S 7))
              (SS (* F F F SC) (* AU AU AU SV))
              (STEMCHR (* * 7))
              (TEXT . KAUFEN)))
```

The various analytic functions thus have uniform access to each other's findings, independent of specific local variables in the program.

An especially significant change in the data structure is its alteration to reflect the structure of German, in grouping of words into phrases and into clause elements. Thus the function NP-SD, which handles noun phrase analysis, will take a constituent article and noun which were originally two WORD entries at the top level of the structure. They will be combined under a new, single NP entry, which also holds information pertaining to the phrase as a whole. A similar structural change is done by COMELTA-SD to group the entire accusative object together into a single element at the top level of the clause. Here is the final data structure from the analysis of the current example, with the structure apparent from indentation as well as parenthetization:

```
(
 (FV-2
  ((OBJECT
    (NP
     ((ARTICLE (WORD EINEN (E I N E N))
               ((ACLASS . 1) (AEND . EN)
                             (ATEXT . EINEN)
                             (ENDING . EN)))
      (NOUN (WORD HUT (H U T))
            ((END . *T*) (SV . U)
                         (NTEXT . HUT)
                         (XGENDER . M)
                         (XCHR ACC S M O))))
     ((GENDER . M) (NUMBER . S)))
    ((PPCREF . PP-8) (CASE . ACC) (CASEWHY OBJECT SPEC)))
   (VERB (WORD KAUFT (K A U F T))
         ((SC F) (SV A U)
                 (CLASS . 1)
                 (END . T)
                 (ENDCHR (3 S 37) (2 P 77))
                 (SECHR (2 P 7) (3 S 7))
                 (SS (* F F F SC) (* AU AU AU SV))
                 (STEMCHR (* * 7))
                 (TEXT . KAUFEN)))
  (SUBJECT (PRONOUN (WORD ER (E R))
                    ((CHAR 3 NOM S M PER) (LEVEL . 4)
                                          (NUMBER . S)
                                          (PERSON . 3)))
           ((PPCREF . PP-3) (CASE . NOM)
                            (CASEWHY SUBJECT SPEC)
                            (PERSON . 3)
```

Phase Two Summary: The clause executive does Phase Two of the match/analysis using the functions described above. RUNWD is called each time for a new word, and then a WD-SDPTR is followed to analyze the associated clause element. This loop keeps up until RUNWD signals FINISHED, generally indicating that the end of the student's sentence has been reached. (In sentences containing relative clauses, FINISHED will be set at the end of each clause.) At this time, all the WDLIST entries should be satisfied, and gone; if not, there are words missing from the clause. At this point, appropriate action could be considered. If a noun is missing, the program may for instance want to check for the possibility of paraphrase by an equivalent pronoun. If so, by simply following the cross-reference pointer of the missing noun to its SDLIST entry, the program can obtain the necessary case and number information to specify the equivalent pronoun; it then generates a PNLIST entry to actually look for the pronoun.

Phase Three:

Next comes Phase Three, at which the PNLIST is run. Each term of the list describes one pronoun. The pronoun search function is prepared to look anywhere in the clause for the pronoun. The major difficulty lies in establishing identity when the clause contains more than one pronoun. Each PNLIST entry has a pointer to its corresponding SDLIST element, just as for the major words on WDLIST; once the pronoun's identity has been firmly established, the link can be followed immediately, as there is no question of possible clause changes.

The technique for finding pronouns is conceptually simple. Each pronoun is represented by a functional entry on the PNLIST. When activated, this function searches all the free words of the clause. Any words which are pronouns are compared with the expected value, using a measure of goodness of fit. An exact match can be claimed immediately. If none of the candidates matches perfectly, the program will choose the "closest" one.*

A tie may occur, if two pronouns lie at the same distance from the one being sought. A tie can exist only if the sentence contains at least two unclaimed pronouns, and the one currently being sought has not been correctly formed. There is a good chance that the other pronoun will be correct; or, failing that, at least not wrong in such a way as to duplicate the current confusion. The strategy, then, calls for the particular search function which has encountered a tie to calmly give way, allowing another PNLIST element to be activated and hopefully claim one of the embarrassing alternatives. The delaying mechanism can be invoked at any stage of the function's operation; and it can be used repeatedly if necessary. (See scheduling discussion, later this section).

---

* See Chapter 7 for a description of the distance metric.

In a pathological case, it may occur that neither PNLIST function is happy with what it sees, and so each is waiting for the other. When this occurs, a modified distance-measuring function is tried in an attempt to break the tie. If that fails, as for instance must happen if the two words are identical, the function arbitrarily picks the candidate situated nearer the end of the clause.

Here is some program output from a simple sentence containing a potential ambiguity in pronoun recognition. The student is supposed to produce "Sie sieht ihn" ("She sees him" or "it"/masculine), but instead uses a neuter pronoun: "Sie sieht es". My comments are in []'s.

```
          ---------- RUN 2 ----------(abbreviated)

THDO: PRONOUN-WD
End of search for pronoun IHN level = 3 FAIR
        [Did not find "ihn", and no way to choose between
        "sie" and "es", so delaying]
THDO: PRONOUN-WD
End of search for pronoun SIE level = 4 GOOD
        [Search for "sie" finds an exact match, so go
        on to SDLIST work:]
THDO: PRONOUN-SD
..........   CASE REPORT FOR SUBJECT: SIE :: CORRECT = NOM

..........   VERB ENDING CHECK .. OK!

THDO: PRONOUN-WD
End of search for pronoun IHN level = 3 FAIR
        [Second try for "ihn". Match results are no better,
        but since "sie" has been claimed already, there is
        no ambiguity and "es" can be claimed.]
THDO: PRONOUN-SD

..........   PN SUBSTITUTION..   Should be (3 ACC S M PER)
           is (WORD ES (E S) 3 ACC S N PER)
..........   CASE REPORT FOR OBJECT: ES :: CORRECT = ACC

..........   Summary Comments
        Clause type correct (FV-2)

          ---------- END OF RUN ----------
```

Many pronouns are associated with more than one grammatical slot (see Chapter 7 for a more complete discussion of this notion). For instance, "es" serves as both nominative and accusative in 3rd person neuter; "sie" serves both plural and singular as well as nominative and accusative. In these cases, the program considers each possible slot as if it were a separate word, so that its final determination specifies the best choice. Of course, if a tie results only from two slots associated with a single word, there is no point in delaying, and the program will proceed directly to a forced choice.

Special consideration is afforded subject pronouns. A brief check on the finite verb ending tells whether the verb agrees with the expected value. (The full verb ending check procedure considers mainly agreement between the verb and the actual subject). If the verb is not what was expected, there is a good chance that the subject pronoun may also be different. In this case, a limit placed on the closeness measure blocks the determination of an exact match, thus forcing consideration of other possibilities. Thus if the verb is singular instead of an expected plural, the program must be prevented from locking onto a "sie"-plural form without also considering its singular representation (and any other nominative pronoun forms that may be present).

In the event of a tie leading to a deadlock, agreement with the actual verb ending counts positively for a subject pronoun candidate.

Here is an example in which the program is looking for "Er kauft es" ("He buys it") and finds, instead, "Sie kaufen es" ("They are buying it"). My comments in []'s.

```
———————— RUN 3 ————————(abbreviated)

THDO: PRONOUN-WD
End of search for pronoun ES level = 4 GOOD
        [success for the object pronoun]
THDO: PRONOUN-SD

..........  CASE REPORT FOR OBJECT: ES :: CORRECT = ACC

THDO: PRONOUN-WD
Subject pronoun: expanding search limits since
verb ending does not match expected subject
End of search for pronoun ER level = 3 FAIR
More than one possibility in search for pronoun ER
        [Only one word, "sie", but several possible forms]
Check with verb ending — 1 choice for subj pronoun
THDO: PRONOUN-SD

..........  PN SUBSTITUTION..   Should be (3 NOM S M PER)
         is ((WORD SIE (S I E)) : (3 NOM P * PER))

..........  CASE REPORT FOR SUBJECT: SIE :: CORRECT = NOM

..........  VERB ENDING CHECK .. OK!

..........  Summary Comments
        Clause type correct (FV-2)

  ———————— END OF RUN ————————
```

<u>Phase Three Summary</u>: At the end of Phase Three, all of the SDLIST entries should have been processed. The previous checks, on WDLIST and PNLIST, ensure that any

entry which is located via a pointer from either of those two will have been done. The additional check at this time allows elements which are purely analytic, without any match entry.

Phase Four:

Phase Four of the analysis consists of checking the word order of the now totally identified clause. The function PHRASEDIVIDER carves out verb/clause type and front field, inner field, and end field. PHRASETYPEREPORT comments on that information and its relation to the expectations as present in the description. The comments produced by both functions are clearly visible in the program output. "Phrase division done ..." is PHRASEDIVIDER, "Summary comments ... " introduces the PHRASETYPEREPORT comments.

Finally, the entire clause is depressed in the data structure holding the student sentence, so the clause appears as a single element at higher levels (as described above in reference to phrases and clause elements, and visible in the example given there).

A bit of careful bookkeeping makes two tricky situations come out right: phrases with more than one major word, and dependent (subordinate) clauses. Within the SDLIST processing, the main scan may have to advance past the major word which provided the initial entry to the

SDLIST: for instance, in a phrase containing two nouns, like "eine Tasse Kaffee" ("a cup of coffee"). In the example, the identification of "Tasse" would take the analysis into the noun phrase; once inside, the analytic entry for the second noun "Kaffee" would be encountered, although it has not yet been seen. RUNWD is called to make the move and look at the next word of the sentence. Note, in the above discussion, that it only sets a cross-reference WD-SD PTR when a word is found, it does not immediately follow that pointer. In the present case, the program notes that the new pointer leads to the element currently being processed, which is fine. If a word should be missing, the RUNWD scan would pick up a major word from another clause element. This change will cause termination of forward scanning for the current clause element, thus meeting an important criteria by not accidentally claiming words belonging to other clause elements. The cross-reference pointer from the new word is saved, of course, and will be followed after the current analysis is finished.

Dependent clauses of all types are handled by a further refinement of the same techniques just described. If the sentence contains a dependent clause, it will be linked into the main clause through either a WDLIST or SDLIST entry, depending on whether it functions as an entire clause element of the main clause or a modifier within some element. Separate WDLIST, SDLIST and PNLIST's are created for constituent members of the dependent clause.

If during its search, RUNWD is unable to identify a word as a either a function word, pronoun or major word from the current clause, it proceeds to check the other WDLIST's. A successful match indicates that a change of clause is needed; the flag NEWCLAUSE is set as indication. The level of each clause, recorded during Phase One, is checked at this time. If the clause in question is not subordinate to the current one, the end of the current clause has been reached (according to the assumption that clauses will be contiguous units); the FINISHED flag will be set to trigger the necessary processing.

The existence of more than one clause in the sentence means that the formal context of the clause itself must be maintained. The function CLAUSER coordinates Phases Two, Three, and Four of the analysis, as well as containing the necessary contextual variables; CLAUSER is simply called recursively for processing subordinate clauses. The separation of WDLIST and SDLIST activity aids this recursion. If the NEWCLAUSE flag is set, the appropriate changes in clause context will take place before the program follows the WD-SDPTR. Thus, even for verbs, which are single-word elements, the analysis work is done by an SD-function. For a clause subordinate to the current one, the context change takes place when the WD-SDPTR is followed; the pointer will lead to a function (CLAUSE-SD) which as part of its action makes a recursive call on the clause executive, with new WDLIST and SDLIST set up for that clause. This change in clause may happen at any level of the program.

Dependent clauses are introduced by one of several types of subordinating words. These words present a bit of a problem to the analyzer. A slight bit of legerdemain makes question words and subordinating words come out right. Grammatically, these definitely indicate the beginning of a new clause.* To keep the program on the track, cognizance should be taken of the clause change. These words do not have enough lexical content, however, to permit unique identification of which clause has appeared if the sentence contains more than one dependent clause. Further compounding the problem, German has a great deal of overlap in the forms taken by question words, subordinating words, demonstrative pronouns, and definite articles. The current implementation for subordinating words uses a WDLIST entry which detects a question word or subordinating word, but the activation of their WD-SDPTR is delayed, as a special case, until some other major word has effected the change of clause. It is not completely clear that such is the best way to proceed; reexamination will come as the program expands to handle more variety in uses of subordinate clauses.

* What does it mean if they are not at the beginning? Some relevant comments are in Chapter 8.

Internal Scheduling

Certain aspects of the internal scheduling algorithms are unusual enough to be of interest. I present them as illustrations of the kinds of flexibility needed to facilitate the analytic task; none are particularly noteworthy on their own. I provide one mechanism for indefinite delay of a task, waiting for some fairly simple condition to be met; a way of postponing a task until a specific time (the end of the clause), at which time it will receive special scheduling; and a way of postponing a task, based on complex conditions, to allow other similar tasks to proceed ahead.

Indefinite delay:

The indefinite delay is handled by having a simple list of tasks to be done. These tasks consist of functions to do various search or checking operations which depend on factors whose time of availability is unpredictable. The task list is checked after every major operation; functions stay until they have done their job and are then deleted. For an example, each verb routine creates a simple task to determine whether its verb is the finite verb or a nonfinite one, a distinction which may depend on the presence of other verbs. (C.f. "He had run", "He ran".) Once that function has been able to make a decision, it may set up a check on subject verb agreement, which in turn might have to wait until the subject has been identified. Some of the position checking functions are currently implemented as delayable tasks, while others have been incorporated in the PHRASEDIVIDER functions. The task list mechanism remains to provide scheduling flexibility, particularly to facilitate the addition of new features which may well find their eventual home elsewhere.

The operation of the task list feature is evident in the verbose example of program output. When a task is first set up, the message "New Tasks:" identifies it. (Parameters are occasionally passed via "TEM-nn" variables to avoid impossibly long printing). I do not show the attempted executions, but when a task is done successfully, the indication "Task just done" appears.

Definite Delay:

The PNLIST provides a somewhat similar way of altering the scheduling, by postponing action until the end of the clause. Any search which must wait until the clause boundaries have been established can be entered on the PNLIST. This feature is currently shared by pronouns and separated separable verb prefixes. The PNLIST is separated from the task list to avoid confusion, and also to provide flexible scheduling (via the complex delay mechanism) when it is finally executed.

Complex Delay:

Within the executive used in following SDLIST pointers (and also for executing the PNLIST) is a mechanism to allow arbitrary delaying by any entry. The SDLIST is arranged according to the structure as given in the original description. Words may be encountered in different order due either to errors in the student sentence or anomalies in the implementation. In either case, it is useful to alter the sequence of execution, specifically by allowing a routine to delay its execution. That routine will be tried again after the next routine has been done; delaying is not limited until there is nothing waiting to be done but routines waiting for each other. As the scheduling changes, it will sometimes be necessary to scan forward in the sentence: RUNWD is itself available at the lower level executive, with appropriate safeguards for handling a change of clause or change of element. The scheduling mechanism, currently used at less than full flexibility, allows a function to decide at any point that a delay is advisable; even changes in the global data structure can be automatically reversed when the delay option is exercised.* I anticipate greater use of the

---

* I wish to acknowledge a certain debt to the M.I.T. Micro-PLANNER system <c.f. ref. 22.3; Winograd> for the notion of complete restoration of the environment when some condition causes a routine to "fail". My implementation, however, is much less complex than Micro-PLANNER.

complex delay feature as more complex noun phrases are implemented. Already, as I have described above, it forms an important part of the pronoun searcher's strategy for handling ambiguous matches.

A slight program trace appears on the full output to indicate the operation of the complex delay executive. The identifier "THDO:" followed by a function name indicates the start of execution of a function which has the right to delay.

PART THREE

Error Analysis

The following sections will discuss the task of analyzing words and structures once they have been located and identified. After an introductory discussion of the general philosophy of errors which I have followed, I shall proceed to a detailed description of the analysis of inflection and word order.

Chapter Six
ERROR ANALYSIS: PHILOSOPHY

The operation of the program can be divided into two areas: a "match" portion (described above) with responsibility for finding words and constructing some of the framework of the sentence; and the "analysis" portion, yet to be discussed, which follows the matching process with checks on the correctness of the various words and other consructions. This checking becomes particularly interesting when there is an error. My approach to grading is more general than a simple determination of right or wrong. Among the many facets are disentanglement of different errors from one another, and extraction of information pertaining to the degree of the student's competence with the subject. I shall expand on both my theory of error analysis and the way it is handled by the program, after briefly reviewing some relevant work by others.

Survey of Other Research

There is little formal precedent for my kind of analytic grading; what discussion exists in the literature is mainly theoretical, not practical.

An experienced teacher can easily grade and analyze a student's answer (though he probably will not be able to explain how he goes about doing so). The teacher will also offer explanations for wrong answers: "that's feminine"; "She used dative instead of accusative there."* In speaking with several experienced German teachers, however, I found that they interpret student errors almost intuitively; that they had considerable difficulty expressing rules of any broad validity. I do not mean to preclude the possibility that more thorough investigation would result in more definite findings; for this research, I have chosen to concentrate on emulating the grading results, indeed on expanding them.

When one refers to the literature on foreign language teaching, the teachers' lack of rigorous approach to error becomes less surprising. To quote one current researcher:

> When one studies the standard works on the teaching of modern languages it comes as a surprise to find how cursorily the authors deal with the question of learners' errors and their correction. It almost seems as if they are dismissed as a matter of no particular importance, as possible annoying, distracting, but inevitable by-products of the process of learning a language about which the teacher should make as little fuss as possible.
>
> <ref. 4.1; Corder, 1967>

---

* The teacher may well draw conclusions which he uses, perhaps subconsciously, to guide his teaching.

A chapter heading from one of those standard works:

> Foreign language habits are formed most effectively by giving the right response, not by making mistakes.
> <ref. 13; Rivers, 1964>

Rivers goes on to quote Politzer:

> The real skill of the teacher lies not in correcting and punishing wrong responses but in creating situations in which the student is induced to respond correctly.
> <ref. 11; 1960>

(Note: I recognize that particular psychological theories of learning have motivated some of the above comments; in the current project, I implicitly subscribe to an alternate view of learning.)

Yet there are examples in the literature of a genuine concern for errors. The language teachers have recently been wooed by Contrastive Analysis, which employs linguistic methods to identify difficulties arising from confusion with the student's native tongue. Independently, researchers studying the speech of young children relied on the presence of errors as evidence that the child was indeed using rules:

> Even children who have never studied the rules of grammar make use of the grammar of the language. This is seen in the mistakes they make. When a child says, <u>He goed</u>, he is forming a "regular" preterite on the pattern: <u>showed</u>, <u>weighed</u>, <u>served</u>: "<u>goed</u>." His error reveals the that he has been applying the pattern even though he is not able to describe it.
> <ref. 8.2; Lado, 1964>

My work attempts to apply similar reasoning to errors found by the program in the student's second language work. I believe that much "erroneous" behavior can be explained as more than just as a simple error, but as a mistaken conception about the scope of a rule, about which rule to apply, or even about the exact formulation of a rule.

Corder, whose introductory remarks I cited above, has some positive contributions of great interest, stating that: "our ingenuity should be concentrated on techniques for dealing with errors after they have occurred." <ref. 4.2; 1967>. He has taken the notion of error analysis from first language learning and applied it to second language work as well. He sets forth the view that the speech of someone learning a second language need not be regarded as improper formulations of that language; rather, the speech can be considered as a language in its own right, complete with a grammar. It is a dialect, of course, sharing a core with the correct or "target" language; it is usually quite individual, and also unstable, as continuing instruction brings it more and more into accord with the target. (Corder calls the dialect an "idiolect".) From this point of view, "mistakes" are not so much to be taken as wrong, but rather as evidence of the particular grammatical system of the "idiolect". Corder challenges the teacher to detect these rules and use the information to guide the student to strengthen or modify his understanding appropriately. <ref. 5; 1971>

I share the general approach propounded by Corder, though I have not carried it as far as he suggests. I very definitely agree that the student's performance reflects the state of his knowledge, and have attempted to implement error diagnostistics on that assumption. (I would note that Corder does not explain how to go about doing what he proposes). That the state of the student's knowledge may be cast as a system of grammatical rules is a conception I am working towards, but have not yet reached.

An Explicit Error Analysis Technique

I should like to elaborate on the theory which I have developed for practical error analysis, before proceeding to a detailed discussion of the actual analysis.

Requirements:

I require several interlocking characteristics in the grading report:

1. Rule-specific reporting. The student is taught specific rules of grammar. Whenever possible, his performance should be related to those rules, preferably on the level of individual rules. For instance, a report should not read "noun form wrong" but something more like

"'-en' ending, required on this type noun for singular accusative and dative, is missing." (Bear in mind, this report is not necessarily intended for the student, but primarily for the teacher).

2. Meaningful error followup. (see, particularly, Corder). A German grading program should be able to interpret a form like "der Buch" as a mistake in gender (substitution of neuter for masculine), or "die Buech" as a good try at a plural form (which should be "die Buecher").

3. Credit for partial correctness (a corollary of the rule-specific requirement). If the student needs to use two rules in a given spot and only does one correctly, he should still be given acknowledgment for that much. This dictum applies at all levels; it is, of course, part of the reason for such detailed analysis of the sentence, the need to separate out errors from correct performance. Even within a single word, the notion of partial correctness may have meaning and should be pursued. Thus in "Der Buch ist alt" ("The book is old"; the German should be "Das Buch ... "), I would like to credit the student for nominative case in the article (probably), even though the form of the word "der" is not totally correct.

4. Strong and weak correct. The rule-specific view of reporting can be extended to include correct as well as incorrect performance. Along this line, it may also be pedagogically useful to know how much of a discrimination

was involved in the student's correct answer. For instance, the article "der" in "Der Mann ist alt" ("The man is old") could be reported directly as a correct application of the rules for indicating gender (masculine) and case (nominative) in the definite article. In many instances, however, there is such overlap of forms that a definite diagnosis is not possible. Thus the article "das", in "Das Buch ist alt" ("The book is old") indicates gender correctly (neuter), but in indicating case does not discriminate between nominative and accusative. I will term this lack of discrimination "weak correct", in that one cannot satisfy a strong assumption of correctness.

Descriptive Framework:

In order to address the question of error in a systematic way, I have set up a descriptive framework describing the major steps in the sequence of rule applications leading to the final form of the word. I can then use this framework to guide the rule-specific reporting. Meaningful error is then defined as a deviation which can be interpreted as a failure or misaplication of a rule somewhere in the framework.

The simplest example of such a framework shows a three-step process: cue — intent — result. For example,

Cue: "____ Mann"

Intent: masculine (nominative)

Result: "der"

For grammatical items this three-step division represents the beginning of a system which takes into account the existence of interesting mental activity between the cue and the final form. In particular, one would like to examine the ways in which the student will activate and apply various grammatical rules.

There is no way to see the middle step, the intent, directly. One can, of course, ask the student direct questions about the rules he is using, and perhaps extract some very useful information. These questions involve interaction with the student, however, and thus represent teaching activity in their own right. As such, one can evaluate the desirability for the direct questioning, but it is outside of the realm of the present project, i.e. extraction of information from the response alone.

If it were possible to see inside of the student's head, the grading method of choice might be:

```
cue —— intent
         |
       result
```

(I.e., major analysis concentrating on the steps from cue to intent, and from there to result.) There would probably also be a check directly from cue to result. All that is available, however, is what is visible, and so the method must be:

cue —— result
(intent inferred)

(I.e., moving directly from cue to result, and inferring from that what the intent might have been.) If the inference is of sufficient quality and detail, then one can reconstruct the more desirable cue - intent step as part of the analysis. Human teachers seem to be able to do this, thus providing the computer with a challenge. Someday, perhaps, psychology will provide a good theory of mental functioning to explain situations like the present one. For the present, a black-box approach seems to provide the best approach. I construct an algorithmic framework which can be presented essentially the same input as is given to the student, and which will produce the same result. From its performance, I then make inferences about the student's command of the subject. To avoid any misunderstanding, the analysis is not done "by synthesis"; the program does not blindly try to generate a form similar to that used by the student and then use its own actions for a diagnostic report. Rather it attempts at each stage to figure out what the student has done, using its information on what is expected, what the final result was, and what kinds of error possibilities exist.

At times the program will be unable to make a firm diagnosis of what the student intended. The source of the diagnostic difficulty may be an unanticipated confusion on the part of the student, which the program could not interpret (other than of course detecting that the form was erroneous). Or, in many instances, a basic ambiguity in German grammar will form an impediment to full interpretation; "weak correct" is a manifestation of this latter problem. In either case, the grader program appears to be in an advantageous position for not only reporting on its findings, but actually pointing to ways in which the curriculum might be altered to elicit a response that would shed light on the particular confusion.

The following sections will describe the specific applications of the general theory of error analysis to the specific problems of inflection and word order.

Chapter Seven

INFLECTIONAL ANALYSIS

Inflection: "The change of form that words undergo to mark such distinctions as those of case, gender, number, tense, person, mood, or voice." <ref. 20; Webster>

German is a highly inflected language; accordingly, the word-match routines have been specifically designed not to be sidetracked by inflectional errors. In this section, I discuss the analytic portion of the program which is responsible for checking inflection.

Terminology

In this discussion, the term "grammatical category" shall denote one of the set of distinctions: case, gender, number, tense, or person. An inflectional change shall be said to "indicate" the relevant category. (These two are standard terms.) Furthermore, I use the term "value" for the particular choice within a category. A noun, for instance, inflects to indicate the grammatical category of number (or more briefly, to "show number"). At a particular instant, the appropriate value in that category might be "singular".

In general, a particular inflectional form will have to reflect more than one grammatical category. I use the term "slot" to denote a specific set of categories and their values, with which is associated a particular inflectional form. (A form may be associated with more than one slot.) For example, articles must show case, number and gender; a slot will be a set of one value for each category, such as: nominative, singular, masculine.

The following abbreviations are used throughout this report, as well as by the program itself:

case:	nom — nominative acc — accusative dat — dative gen — genitive (not implemented)
number:	s — singular p — plural
gender:	m — masculine n — neuter f — feminine

In any category, "*" may occur to indicate an absence of discrimination: "*" will match any value.

The slot associated with a particular inflectional form may also be called its "characteristic".

On occasion, I may regard the indication of a category as a "constraint" on the word in question, and speak of satisfying the constraint.

Framework for Inflectional Analysis

The previous discussion on error philosophy and rule framework applies most strongly to inflection. Accordingly, I have a framework setting forth four major steps:

1. What grammatical categories must be indicated?
2. For each category, where is its value determined?
3. For each category, what is that value?
4. What is the proper orthographic form to show all the values?

(Note that "intent" now comprises steps 1, 2 and 3; the "result" comes from step 4.)

An example may be useful. (for simplicity, I will omit the category Number.) Consider the definite article, "den", in "Ich sehe den Wagen" ("I see the car").

1. Must show: case, gender
2. Governors: case: clause syntax, verb
   gender: from noun "Wagen"
3. Values: case: accusative
   gender: masculine
4. Form: "den"

Step 1 is so basic it is easily forgotten. The computer, of course, needs explicit instruction on what is required. Generally the student will also have been told similar rules, which exist in his grammar book even if not in the material as presented by the course. If he forgets the inflection completely, the error is comparatively easy to spot: an attributive adjective will appear without an ending, or a verb in the infinitive form. More difficult is the situation in which one grammatical category was forgotten, as words which indicate more than one category usually do so with a single form, as in the above example "den". (The alternative is illustrated by the English pattern: man - men - man's - men's which shows plural and possessive independently.) Certain possible errors may suggest a step 1 failure. The definite article, for example, is often introduced as showing only gender, not case, using its nominative form.* Thus the student who forgets to indicate case might well just reproduce this nominative form; occurring where the correct case is dative or genitive, this might point to the step 1 error.

Step 2 constitutes the formal statement of the "agreement" requirements (which clearly can also be phrased as a requirement that both words indicate the same values for the relevant category.) Viewed from a slightly different angle, this states the need for internal consistency. The verb, for instance, must agree with the subject of the sentence as written; if that subject is not the expected one, the verb still should show the internal

* compare the following statement from the Reference Grammar: "The plural article for all three genders is 'die'." <ref. 9.2; section C101>

agreement. Similarly, if a preposition has been changed from the expected one, the student must apply the case requirements for the preposition he has actually used. Of course, he should be credited for following these precepts.

When a substitution by the student results in a change from the expected value of the word governing a particular inflection, the grader must be sensitive to both the expected word and the one actually used. Agreement in form with the expected word is a kind of step 2 error: the inflection rule requires agreement with the actual word. Possibly, however, the substitution is only in the form of the governing word, not its intent (i.e., a step 4 error on that word); the student might then be using his intended value as the source for the current inflection. Such guessing can become very tricky, but the expected value represents one good hypothesis about what might be intended and is well worth checking. Another possibility, producing much the same result, is that the student learned the inflectional pattern by rote, but has forgotten a middle step, the form of the word which actually governs a part of the pattern. For example, he may recall that "to count on" — "rechnen mit" — requires a dative object, and use dative even if he forgets which preposition to use and substitutes one which itself requires accusative.

There are not too many instances within the language itself in which the student can be expected to make a wrong choice of governor. Several distinct possibilities which do exist are worth noting briefly.

—The gender of a compound noun is determined by the last component noun;

—The verb in the form "Das sind xxx" ("That is xxx") agrees in number with the complement, not the normally-singular subject "das";

—For certain neuter nouns which have "natural gender", such as "das Maedchen", pronouns will agree either with the grammatical gender or the natural gender, depending on the physical distance between noun and pronoun.

In all of these cases, the program can certainly be made sensitive to the possibility of step 2 errors.

Having reached step 3, the student just may not know which value is appropriate, particularly when that happens to be a n gender. I find the distinction between steps 2 and 3 intriguing due to the prospect of giving the student credit for an inflection even if he errs at the point of determining the value for a particular grammatical category. Notice that an error in noun gender is really a problem with the noun, even though it happens to show only in the formation of the article.

At steps 3 and 4, the student must command a fairly large amount of information to perform correctly. In some instances, the information constitutes a closed set, albeit a moderately large one (e.g., adjective endings, irregular verbs). In others, the complexity of each inflection may be less, but the size of the set is unlimited (e.g., nouns, gender and formation of plural). Lots of things to remember means, generally, lots of possible errors. I have accordingly devoted quite a bit of effort to handling step 3 and 4 errors. At the same time, I must add that much of the information the student needs is merely factual, learned by rote, which makes for rather uninteresting errors. (I'd much prefer to chase an improperly learned rule.) Even uninteresting errors need to be caught.

The detailed analysis of errors at steps 3 and 4 has two dividends beyond the basic necessity of error detection. Both lie in the direction of sharpening the precision with which the program can diagnose just what the student knows.

A) Proper object of error:

A real step 3 error belongs with the word which is the source of the constraint as much as with the word showing the improper inflection (unless, of course, they are one and the same word). Usually, the source word will not itself inflect to indicate the category in question.

Thus a student who thinks that "Wagen" ("car") is feminine will say "die Wagen" instead of "der Wagen" in the nominative singular. As the error comes from the noun, it will also affect other constuctions like the indefinite article and adjective endings as in "eine rote Wagen" ("a red car"), or pronoun gender as in "Hier ist eine Wagen. Sie ist huebsch" ("Here is a car. It is pretty").

On the other hand, the student's form "die Wagen" might be due to a step 4 error; i.e., not knowing the correct form of the article. Then the error indeed lies with the article, not the noun.

In an instructional system that tries to respond to the individual characteristics of each student, correct placement of credit for errors carries major pedagogical implications.

B) Disentanglement to find out what is correct:

As noted above, the German inflectional system usually combines all the constraints into a single form at step 4. There is no quality of the nominative masculine ending which expresses the nominative as distinct from masculine in a way that might be expected to carry through to, say, a nominative feminine form. A mistake in indicating one grammatical category will in general produce a form in which the correct performance in other categories is not immediately obvious, and it is only through systematic analysis of the possible steps in the production of the form that the grader can properly assign credit.

For instance, the definite article:

<dat p m> — "den"

<acc p m> — "die"

<dat s m> — "dem"

A "dem" where dative plural is expected will be diagnosed as proper case, wrong number; "die" in the same slot, as correct number but wrong case. But these diagnoses come not from the orthographic form itself, but from conclusions gained by following back to a point at which the various grammatical categories can be separated. In this circumstance, establishing meaningful interpretations of errors serves a more important function than merely indicating how the student failed in one category; it allows recognition of a simultaneous correct performance in another category.

Unfortunately, things are not quite so simple. There is no a priori way of determining exactly what the student had in mind (see above, Chapter 6). In particular, any given error might be due to step 4 (wrong form) just as easily as step 3 (wrong value) or even lower. Each interaction with the student provides a little window into the state of his mind. Someday a program might exist which would record all performance and thus be able to say, yes, Bill thinks that "Wagen" is feminine, so expect step 3 errors and forms like "Hier ist die Wagen" ("Here is the car"; correct German would be "...der Wagen"). Some of this student history might come from grammar drills, whose direct questions provide small but clear windows. The grader's analytic capabilities might well be put to use even before presenting an exercise, to determine whether it would be likely to provide the desired information or just lead to a diagnositic ambiguity.

There are quite a few possibilities relating to errors which can be covered already, i.e., without recourse to the individual student performance history. Some are from general errors as observed from traditional teaching, others from logical analysis of the grammar and the rule-application framework. A [incorrect] form will either be a German inflectional pattern or it will not. In the latter, probably less common case, there is by definition a step 4 error: the form is not correct. Unfortunately, until the program has history information available, nothing more can be extracted from an unrecognized form; it is just wrong.

If the inflectional form is recognized, perhaps a simple explanation can be found to explain an error. Much more detail on this procedure follows below. General principles include the following:

a) If the meaningful explanation postulates an intent too far away from the expected values, I would incline away from a step 3 explanation, probably towards step 4 except for the special circumstances discussed above. I'd not expect, for instance, a genitive form where nominative belongs.

b) If the governor of the inflection can be identified as likely to cause error, a bias can be introduced in favor of that (step 3) explanation. English-speaking students, for instance, tend to be quite prone to forget noun genders.

c) If other words involved in an inflectional group are themselves inflected, the extra information can be used to throw light on what the student had in mind. This situation rapidly becomes quite complex.. see discussion below.

There exists one very bad complication: the incompleteness of the inflectional system. (Incomplete in a mathematical sense.) Some forms serve for more than one value, which results in a weak indication of the category in question. (e.g., only masculine articles distinguish between nominative and accusative case.) Also some orthographic forms appear in several, apparently unrelated slots. ("Der", for instance, serves both <nom s m> and <dat s f>.) The result is an inherent ambiguity in the process of reversing the formation rules (step 4). I have worked out various ways of coping with the ambiguities, ways which are valuable as illustrations even though lacking the specific student data which will clearly be needed to achieve the full potential of the grader system.

Determining What the Student Intended
Three Methods

I have explored a number of approaches to the problem of arriving at a reasonable guess as to the student's intent. Two methods have been implemented; a third proposal may be even better. Briefly, the first involves built-in tables giving intent hypotheses for all combinations of responses and expected characteristics; the second method is computational, choosing among alternative explanations on the basis of their "distance" from the expectation. The third is also computational, working with conditional probabilities related to the student's performance.

1. Tabular Method:

The first approach, the tabular, is currently implemented for article-noun inflection analysis; it is responsible for the comments in the sample runs. This approach employs a direct table lookup to produce an estimate of what the student intended. The two input parameters for the table are the expected slot and the observed form. For each combination, the table gives one or more slots, possibilites that the student might have intended. With each slot is associated an indication of the level of confidence I place in the particular prediction. For example, consider the article ending "-er", which is correct for two case-gender slots, <nom s m> and <dat s f>. If this ending occurs in a situation in which the expected case is accusative, the student may have one of three slots in mind: <nom s m>, <dat s f>, or perhaps even <acc s m>. Although the latter is never correct for "-er", it would be consistent with the pattern of neuter and feminine endings.

For a masculine singular noun, the table entries are as follows:

<nom s m> - "medium"
<acc s m> - "high"
<dat s f> - "low"

where "medium", etc., are the level of confidence indicators for each slot, given on a rough scale.

If the correct noun gender is feminine, still expecting accusative, the entries are different:

<nom s m> - "none"
<acc s m> - "med"
<dat s f> - "high"

For masculine (expected), then, the best guess is that the student intended <acc s m>, with "high" confidence in the guess. Interpreting according to the general four-step scheme for inflection, this represents correct intent but incorrect form (step 4 error). For a feminine noun, a "high" confidence guess of <dat s f> means an error in intent (step 3) combined with a correct form.

A complete presentation of the table, with further explanation, comprises appendix B.

<u>Evaluation of method 1</u>: As currently implemented, the tabular method has both good and bad qualities. I desired a demonstration of how intent-guessing could aid the grading operation. The table has been successful in providing plausible guesses to illustrate both parts of the concept: immediate diagnosis and the further analysis possible once the data is available.

The table has a very ad hoc quality; grammatical facts aside, all the plausible alternatives and particularly the levels of confidence are my own creation. Actual values are necessary for the demonstration; presumably the various entries can be adjusted as experience is gained about what students actually do.

The tabular approach has the advantage of providing very specific control of the various alternatives. Since each entry is given directly and individually, there is no need to juggle numbers in order to have two interacting slots both come out correctly. (See, particularly, the following method.) This specificity is at the same time a most severe disadvantage. The table does not contain any direct relation to general instructional parameters such as "student tends to use feminine instead of masculine"; thus there is no obvious way for properly incorporating feedback. Furthermore, the table is noncomputational, leaving no direct way of expanding it to cover other variables, such as word type, or to handle interaction with other in[illegible]cted words.

To meet some of these objections, the table was designed to provide more than one possiblity (intent-guess) per entry, with varying probability/confidence levels. In order to reflect the influence of other criteria, the confidence levels could be systematically altered. The table itself specifies some case-gender entries merely but referring the other entries with directions to diminish the confidence levels. No outside use has yet occurred; it is the nature of the table to want to specify everything individually rather than by broad classes.

Another shortcoming of the table stems again from its very specificity: information beyond that explicitly present versus just not accessable. The lack is particularly galling when the variables I need were themselves involved in my deliberations when setting up the table. The data in the table does not suffice for a determination of whether a form is indeed correct; that information would have to be added, albeit at minimal cost. Some distinction between strong and weak correct has been incorporated in the format of the guess specifications. If a form covers two values of a category, both are included in the slot specification, as in <(nom acc) s f> for "die". But that information still applies only to the intent-guess, not to what actually is correct in German grammar. The latter could also be added, or even obtained computationally, once the correctness information is added.

Yet another frustration occurs when the diagnosis is either ambiguous or confused. The grading program ought to be able to look at what has confused it, and communicate that information to guide the teacher in producing a modification of the exercise which will elict a more informative response. Again, the most convenient course is to expand the table to include the necessary information explicitly. Philosophically, I disagree with the general approach of adding everything directly to the table. If a systematic procedure is involved, better that the program should be expanded to include the pertinent algorithm.

2. Distance Computation Method:

The second method, the computation of a distance metric, is currently employed during the pronoun search. It was also used at one time for article-noun inflection and a modified form is still operational for verb ending checks. This technique involves generating a metric in the space of grammatical alternatives, so that a "closest" alternative can be chosen. When applied to a single variable, this technique is very straightforward; exact matches are closer than mismatches. Thus, to continue the above examples using article inflection, the student might provide "das" when expected to produce a nominative neuter form. This form is correct for both nominative and accusative; the former, being correct, is closer and thus chosen.

A finer ranking is often feasible, even with only one variable. For instance, the student may use the same "das" when dative case was required. Now neither case matches exactly, but dative is "closer" to accusative than it is to nominative (both being object cases), and so a decision (guess) could be made for the accusative.

This method leads to a sharp dichotomy, which was evident in the last example. The choice of the closest of the characteristics associated wih the actual response (accusative, above) provides a guess of what the student intended assuming that no step 4 error exists (i.e., that the orthographic form adequately reflects the intent). The characteristics associated with the expectation represent a viable alternate guess, under the assumption that there is an error in the form itself. The program is in a somewhat embarrassing situation, with no way to make the choice and, moreover, no way to find a middle ground. At one stage of experimentation, I simply had the program carry along both values for further processing.

When two (or more) variables are involved, the distance metric becomes more complex and also more interesting. In simple examples, the number of changes plays a principal role: the distance is directly proportional to the number of variables (or categories) involved whose values differ in the two terms being compared. Thus, with an expectation of <dat n> and an actual form "der" – <nom m> & <dat f>, the analysis is:

<dat n> to <nom m> : 2 changes, case and gender

<dat n> to <dat f> : 1 change, gender

and thus the second is closer. The plot thickens if there is more than one possibility with the same number of changes. Change the example to an expected value of <dat m>:

<dat m> to <nom m> : 1 change, case

<dat m> to <dat f> : 1 change, gender

Here a system of relative weights is needed to complete the choice. Case is probably more significant than gender, so it can be given more weight (i.e., a greater distance is involved if case must be changed). Then the guessed intent in the last example will be <dat f>, chosen because it is closer than <nom m>.

This method is quite appealing. It is simple, yet easily extensible to handle additional grammatical categories; it is computational; and it makes use of general observations of grammatical performance. Not surprisingly, this is the technique mentioned in the literature in the few instances where error interpretation has been considered. Halliday has this to say:

> It is important to note that a given error can be often be described in two or three ways, to each of which corresponds a different step that could be taken to correct it. For example, 'he asked a new book' could be corrected either to 'he asked for a new

> book' or 'he requested a new book'; these will lead to two different analyses of the error, in this case as it happens at different levels: the one grammatical, the other lexical. Both analyses are valid. ... Descriptively, the analysis which yields a simpler correction will be preferred. 'Asked for' and 'requested' are each minimal corrections, in the sense that each requires only one step; but 'requested' involves a change of register and might therefore be inappropriate.
>
> <ref. 6; 1964>

Evaluation of method 2: Neither Halliday nor anyone else gives any more specific guidelines on the weighting of different errors. In trying to set up a working version of the scheme, I encountered some severe shortcomings. I have already mentioned the lack of a middle ground, a compromise between the two extremes of expected and actual values. By associating initial weights with each actual ending/value pair, it should be possible to introduce plausible error alternatives into the basic data set, just as in the tabular method; the initial weights would be such as to place these alternatives at a disadvantage except under certain error combinations.

In the end, the sheer weight of all the different parameters involved with nouns and articles caused the procedure to become unmanageable. As the number of variables increases, so does the necessity of retaining unattractive terms at intermediate steps in the computation on the chance that they could later become significant.

I found the algorithm required an excessive amount of thrashing to accomplish even minimal goals. Either there would be too many final possibilities or one that I was interested in was below the cut-off point. Of course, this behavior could be due to bad tuning, but then tuning itself represents a major problem. The particular way in which ending patterns interlock is such that much of the supposedly general analysis is really oriented towards special cases. (For instance, "der" is the only form appearing in two fully separate entries, <nom m> and <dat f>). The relative weights used for the distance calculations, which were initially ad hoc values, had to be juggled time and again to get the special cases to turn out as desired. With each adjustment, the parameters moved further from the world of the student.

<u>Pronouns</u>: For pronoun checking, a more restricted domain has led to a moderately successful implementation of this second method. The distance metric plays a vital role in the actual match/identification step, enabling the search logic to chose which of several candidates is closer to its target. (For a fuller discussion of the search strategy, see Chapter 5.)

Pronouns indicate five grammatical categories: person, case, number, gender, and type (the latter's values including personal, familiar-personal, and demonstrative).

The algorithm assigns a metric value from 1 to 4 to a match between two slots. Comparison is done left to right in the order given above, discontinuing when a mismatch occurs. Specific value assignments are:

1 — when nothing matches (mismatch on person);

2 — when only person matches (mismatch on case);

3 — when person and case are okay, but number, gender, or type do not match; 3 is also assigned if the case mismatch involves an interchange of accusative and dative;

4 — when all five categories match correctly.

Under some circumstances, a second metric is needed to try to resolve a distance ambiguity, i.e., when two comparisons yield the same value by the previous algorithm. The second metric is arrived at by checking all five categories and counting the number of mismatches without using any weighting function.

<u>Verbs</u>: For checking verb inflections, the simple comparision scheme has been quite satisfactory. Basically a yes-no technique, it checks two slots for intersecting specifications and returns either nil or the intersection value. A fuller discussion of its use will be found a bit later on.

3. Probability Computation Method:

The third method, using probabilities, represents an attempt to incorporate the better points of the two previous methods. It has be in fair detail, but has not yet been program e following discussion unfortunately does not have the force of experience other than a little hand simulation. The technique is grounded in the simple three-step theory of how the student goes about generating an inflectional form: he receives a cue (expected value), derives an intended value, and from that produces the actual form. With each step, I associate a probability measure: for a specific cue, the probability of a specific intent; given a specific intent, the probability that a certain ending will result. Several useful predictions result from simple combinations of these two basic probabilities.

Let:

— P(intent[i]) be the probability that the student intends a case-number-gender slot I;

— P(form[j] | intent[i]) that he produces form[j] given that he intends slot I.

The product of the two gives:

— P(form[j] & intent[i]), the probability that he will produce the right form for the right reason (intent):

P(form[j] & intent[i]) = P(form[j]|intent[i]) * P(intent[i])

The probability of producing a form regardless of intent is the sum of all the applicable terms:

P(form[j]) = sum(all i) {P(form[j]|intent[i]) * P(intent[i])}

Note that the expected case does not follow any probability distribution, and so the P(intent[i]) is not strictly a conditional probability even though referred to a given expectation. Different values will be needed, though, for each expected slot. Additional comments on this point will be found in appendix C, along with a complete listing of suggested probabilities.

In this terminology, an intent-guess corresponds to an intent[i] given both an expectation and an observed form[j]. The probability expression is P(intent[j] | form[i]). It can be calculated (with the available information) using Bayes Law:

$$P(\text{intent}[i] \mid \text{form}[j]) = \frac{P(\text{form}[j] \mid \text{intent}[i]) * P(\text{intent}[i])}{\text{sum (all } k)\ \{P(\text{form}[j] \mid \text{intent}[k]) * P(\text{intent}[k]\}}$$

The program need only make the computation for all relevant intent[i] slots, and chose the one with the highest probability. As a given form[j] will not have any signficant probability of occurrence in any but a few slots, the calculations need not be excessively long.

Some examples may be helpful.

P(intent):

If the desired case is nominative, one would hope that the student would intend just that, but there is also some chance that he will chose another case. Assigning specific probabilities his situation, P(intent), gives the following table. To in forming a concrete demonstration, I have also assigned, ad hoc, numerical values to each probability.

Expect:	nominative	
Intent:	nominative,	P(nomnom) = .92
	accusative,	P(accnom) = .05
	dative,	P(datnom) = .03

P(form[j] | intent[i]):

Given an intent, what is the probability that the student will produce a given ending? (This is a true conditional probability.) Again associating specific values for clarity, here is a partial table for P(form[j] | intent[i]):

Intent:	nominative masculine	
Forms:	"-en",	p(en|nom-m) = .85
	"-em",	P(em|nom-m) = .05
	"-er",	P(er|nom-m) = .10

Note that anticipated errors must be included explicitly in the forms table.

P(form[j]):

By combining the two above probabilities, it is possible to predict whether the student will come up with a particular ending for a particular reason:

$$P(form[j]\ \&\ intent[i]) = P(form[j]\ |\ intent[i]) * P(intent[i])$$

The pro[illegible] can be expressed for any situation. Thus, for instance, if the expectation were accusative masculine, the correct ending would be "-en"; the chance that the student would in fact give that ending is mainly:

$$\begin{aligned} P(en\ \&\ acc\text{-}m) &= P(en|acc\text{-}m) * P(acc) \\ &= .85 * .90 \\ &= .765 \end{aligned}$$

The student's intent is not visible in his final performance, of course, so the probability of actually producing the correct form will include terms accounting for other ways it can be produced, raising the probability figure somewhat.

P(intent[i] | form[j]):

The analytic problem, which is my main interest here, is just the reverse of generation. The expectation remains and, in addition, the student has actually done something. The task is to arrive at an estimation of likely intent. The Bayes Law formula above permits a calculation of exactly what is needed.

Consider "-er" used with an expectation of <acc m>. The conditional probabilities for "-er" in the masculine are:

$$P(er \mid nom\text{-}m) = .95$$
$$P(er \mid acc\text{-}m) = .10$$
$$P(er \mid dat\text{-}m) = .05$$

and the direct probabilities for each of the cases involved, with <acc> intent, are:

$$P(nomacc) = .04$$
$$P(accacc) = .90$$
$$P(datacc) = .06$$

The calculation of intent is straightforward. First the numerator portion of the formula is:

$$P(intent[i] \mid form[j]) = P(form[j] \mid intent[i]) * P(intent[i])$$

$$P(nom\text{-}m \mid er) = P(er \mid nom\text{-}m) * P(nomacc) = .95 * .04 = .0380$$
$$P(acc\text{-}m \mid er) = P(er \mid acc\text{-}m) * P(accacc) = .10 * .90 = .0900$$
$$P(dat\text{-}m \mid er) = P(er \mid dat\text{-}m) * P(datacc) = .05 * .06 = .0030$$
$$\text{sum} = .1310$$

The denominator of the Bayes Law formula is a normalizing factor, the sum of all the numerator terms. Carrying out the normalization (divide each by 0.1310) gives:

$$P(nom\text{-}m \mid er) = .29$$
$$P(acc\text{-}m \mid er) = .69$$
$$P(dat\text{-}m \mid er) = .02$$

Therefore the best guess is that the student intended <acc m>, with <nom m> considered somewhat less probable. (The full calculation would show terms for gender and number, and include a figure for <dat f>. See appendix C for more details.)

<u>Evaluation of method 3</u>: The numbers are still drawn from a hat and are subject (at this stage) to minor tweaking to make things come out right. Still they seem more closely related to plausible student behavior than were the numbers in other methods. Moreover, their scope is wider than just the intent-guessing problem. Of general applicability, the probability estimates may be used in selection of curriculum as well as in grading; performance data from any source can be directly related in adjustments to the estimates. The nature of the variables is such that modification through feedback should be relatively straightforward. Similarly, it will be easy to introduce a bias to reflect a changing curricular environment. For instance, a discussion of dative prepositions may temporarily change the relative accusative/dative bias, which just means incrementing the dative probabilities at the expense of those predicting accusative.

<u>Additional Considerations</u>: Another major benefit inherent in the probability technique relates to the question of ambiguity of forms: the weak correct performance criterion. To review, this refers to a situation in which a particular inflectional form correctly expresses more than one value of a particular grammatical category. Clearly there is no way to be certain which value the student had in mind when he used the form; the notion "weak correct" indicates that he is not wrong, but

may not be totally right either. An extension of the probability method provides a rather natural way to catch these finer points. The extension is quite simple; it centers on a consideration of the symbolic, as well as numerical, probabilities.

A weak correct situation will be easy to identify. There will be, of course, the most probable value: the program's guess at the student's intent. There will be a second term, another possibility, with two characteristics:

1. the ending probability, P(ending|intent), will be fairly high, indicating a correct entry; and
2. the symbolic expression of probabilities will differ in only one term from the expression for the selected entry.

The ambiguity responsible for the weakness is precisely in the grammatical category associated with the differing term. An example may help clarify the operation:

> "-e" is correct for feminine singular, both nominative and accusative case.

If the expected case is nominative, the ending probability terms P(e|nom-f) and P(e|acc-f) are both near 1.0, as they represent correct performance:

<nom s f>: P(e|nom-f) * P(nomnom) * P(samegen) * P(samenum)
<acc s f>: P(e|acc-f) * P(accnom) * P(samegen) * P(samenum)

—only the case terms are different.

If the expected case is accusative, exactly the same ending probabilities appear. Again, only the case terms will be different, but this time such as to bias the choice towards accusative:

<nom s f>: P(e|nom-f) * P(nomacc) * P(samegen) * P(samenum)
<acc s f>: P(e|acc-f) * P(accacc) * P(samegen) * P(samenum)

Since P(accacc) and P(nomnom) reflect correct choices, they will greatly exceed their counterparts P(nomacc) and P(accnom) which do not. As the case terms represent the only difference in the second part of the expression, the numerical values will favor nominative if nominative is expected, accusative if that is expected. The change is due entirely to the changed expectation; the performance is precisely weak correct and cannot be used in feedback as evidence that the student can correctly make the particular distinction involved.

The full strategy for handling grammatical ambiguity includes not only detection, as just outlined, but a mechanism for suggesting a way out of the difficulty. A change in the expected gender, case or number will often result in an exercise in which the desired grammatical distinction can be observed. The full algorithm performs both steps, detection and recommendation.

Throughout the following description, I will interpose examples in brackets, taken from the following data:

"-e" is correct for feminine singular, both nominative and accusative case.

If the expected case is nominative:

<nom s f>: P(e|nom-f) * P(nomnom) * P(samegen) * P(samenum)
<acc s f>: P(e|acc-f) * P(accnom) * P(samegen) * P(samenum)

1. Represent probabilities both symbolically and numerically.

2. Use numerical values to select most probable alternative.
[<nom s f> will win]

3. Examine other possibilities for both:

3a. high probability for ending itself
[P(e|acc-f) = .95]

3b. only one term of symbolic expression for P(intent) different from expression for most probable alternative.
[P(nomnom) vs. P(accnom)]

4. Consider the grammatical category represented by the differing terms: this will show the location of the ambiguity.
[case: could be nominative or accusative]

(determination of a better exercise)

5. take the specification for the most probable alternative, and change the value for a grammatical category other than the one involved in the ambiguity. This forms a new specification.
[<nom s f> : try changing gender to "m", giving <nom s m>]

6. Find the most probable ending listed under the new specification.
["-er", Probability .95]

7. Alter the new specification in exactly the same way as involved in the original ambiguity.
[<nom s m> : becomes <acc s m>]

8. Examine the probability for the ending of step 6 given [illegible]cification arri[illegible] at in step 7.
[illegible]er|acc-m) = .10

8a. If it is high, then the same ambiguity remains; return to step 5, making a different change.
[would happen in neuter, as P(es|acc-n) = .95]

8b. If the ending probability is not high, then the algorithm is successful. A new exercise conforming with the new specification will yield a situation in which the current ambiguity in the student's performance can be cleared up.
[so next time try using a masculine noun.]

General evaluation of the three methods:

In conclusion, I feel that the probability method offers great promise of being the best of the three methods considered. Accordingly, I have placed its implementation high of my list of needed improvements to the program.

The following topics in inflectional analysis remain to be discussed, in the order given:

— Word "type" as an additional grammatical category
— Multiple-word inflectional patterns
— Verb ending checks (which provide a good illustration of the other two).

Word "Type" as an Additional Grammatical Category

A basic addition to the program's representational scheme for inflection is motivated by the existence of varying patterns of endings for some word classes. The article "der", as has been noted previously, represents both the <nom s m> and <dat s f> slots. However, its parent, the article/adjective ending "-er" is limited to "dies- words" for the <nom s m> slot; "ein- words" normally take a null ending in that slot. To make the representational scheme uniform, I have found it useful to expand the notion of grammatical category to include, as a category much like case or gender, the "type" or "class" of the word. By so doing, I am able to represent an inflectional form inflectional form purely by the values of the grammatical categories it indicates without having to append special conditions. Thus, in the previous example, the "-er" article/adjective ending could be considered as a possible representation of nominative masculine for a particular type of word. Its use on an "ein- word" could be considered by the error analyzer as a possible <nom s m> form with an error in word type, using very similar logic to that needed to detect a mismatch of case or gender.

One must, of course, be careful not to establish categories which do not have a parallel in what the student knows, or the computer will be mislead into analyzing patterns which do not exist. The creation of new categories is done when necessary to allow the representation of slots to be clean and orderly, but also with care to see how the addition relates back to what the student is taught. Following that guideline, I have set up 6 additional categories which I shall describe area by area. The specific identifying numbers accompanying the description are mainly for internal reference of the program.*

Articles:

For articles, a Type category to distinguish between "ein-words" and "dies-words":

1: ein-words, no ending in nominative masculine, nominative and accusative neuter.

2: dies-words, "-er" in nominative masculine, "-es" in nominative and accusative neuter

3: either at student option

Complications (not yet implemented, except as noted):

"-es" becomes effectively "-as" for the definite article <nom s n> and <acc s n> ("das" instead of "des");

"ein-" takes type-2 endings in the special form "was fuer ein-";

"ein-" has no plural form (implemented);

adjectives use many of the same endings, but also have another set, the "weak" endings, with rather complicated rules for deciding which set to use.

---

* The actual numbers are really octal, corresponding to single bits, to allow logical-"or" conditions on various tests and specifications.

Nouns:

For nouns, a Type category to distinguish the varying inflectional patterns, chiefly in formation of the plural.

Some texts classify nouns according to the particular ending used for the plural as a means to remembering which ending to use. That particular classification is of no use here; in general the choice of ending is essentially arbitrary and best presented to be learned by rote. The program may well note that the student has selected a wrong alternative plural ending as distinguished from a nonexistent one, but that diagnosis does not require a grammatical category.

There is an interesting interaction, however, between the particular ending and the specific kinds of inflectional changes which will occur. For example, most nouns must add an "-n" ending in the dative plural; but a noun whose plural is formed with an "-n" will not show further inflection for the dative plural. Kufner distinguishes some noun classes on these criteria <ref. 7.1>. Even though the distinction does not appear explicitly in common texts, it fits right in with the way I have employed grammatical categories. In order to know which inflectional slots to associate with a particular form, the program must know which inflectional pattern is involved (and worry about whether the student knows it too). Accordingly, I have a type category for nouns, with a six-way division:

(1&2: Nouns which do not inflect at all to indicate plural, neither by umlauting the stem vowel nor adding an ending. This class, incidentally, does not include any feminine nouns; since the article/adjective endings are the same for feminine and plural, there would be no way to distinguish the two if the noun itself did not inflect.)

1: No inflection for nominative plural, but add an "-(e)n" for dative plural.

2: No inflection for nominative plural, and stem ends in "-n" already so no dative plural change either.

4: Nouns whose only change for plural is to add an umlaut, and whose stems end in "-n" so there is no dative plural indication.

10: Nouns using "-(e)s" to form the plural; these nouns also do not distinguish dative plural, and use the same ending for genitive singular unless feminine.

20: Nouns using "-(e)n" to form the plural, do not distinguish dative plural.

40: [Masculine] nouns forming their plural with "-(e)n" which use that form also for accusative, dative and genitive singular.

Complications (not yet programmed):

Nouns with very irregular plurals;

Compounds with "-mann" forming plural with either "maenner" or "leute";

Some nouns formed from adjectives, which use adjective endings.

Verbs:

For verbs, a Type category related to the kind of irregularity.

Traditional grammars list groups of verbs with similar vowel changes, the seven "ablaut classes". These could be easily included in my program if an instructor desired. However, the emphasis in teaching seems best placed on the individual verb's form rather than its ablaut class, particularly since the classes are riddled with exceptions: so it does not seem productive to use these classes in the grader program.

Definite inflectional patterns do exist, depending on the particular degree of irregularity of the verb. For instance, some irregular verbs indicate third person singular present with a changed stem vowel, others do not. This type of pattern, even though not taught explicitly, fits precisely the extended notion of grammatical category.

(1,2,4: no vowel change in present tense.)

1: regular verbs

2: weak verbs with irregularities

4: strong verbs

(10,20,40: vowel change in 2nd and 3rd person singular present)

10: weak irregular verbs (especially "haben")

20: normal strong verbs

40: modal auxilliaries, which use the irregular form in the 1st person singular as well, and have a null ending for both 1st and 3rd person singular present (also "wissen")

Complications (not yet programmed):

"sein" ("to be") fits no pattern at all in the present;

some stems ending in "-st" and "-s" have peculiarites in 2nd and 3rd person present, adding only part of the "-st" ending or none at all;

and many more.

Note:

At this time, only present tense has been implemented, so the category has only been partly laid out. In particular, no distinction yet exists between classes 2 and 4, as their irregularities appear only in the past participle and past tense. Other provisions will be needed to handle certain characteristics of the past participle.

Evaluation of the Type Category:

The patterns described above have an existence independent of the means used to describe them. The particular method employed for checking inflections will determine whether the type category is appropriate. For method 1, the tabular approach, noun inflectional patterns are a very real perturbation. The table will probably need to separate entries for the noun type as given. Method 2, the distance metric, assigned a value to type errors. It proved difficult, however, to work the desired degree of precision into the weighting scheme; emphasis on proper type seems to vary from slot to slot, perhaps depending on whether the class requires a positive inflectional indication or simply the lack of one.

For method 3, probabilities, I have not yet done the necessary analysis. I believe that specific probabilities can be introduced to get the kind of weighting I needed but could not get with method 2. For verb inflection checking, the typing system have been most useful. It allows a facile diagnosis of situations such as the conjugation of a verb on a pattern which would be correct if the verb were [were not] irregular.

Multiple-Word Inflectional Patterns

Quite frequently, the inflectional check consists not just of one actual form versus the expected characteristic, but of several forms all showing some inflection. For example, even a simple noun phrase will have inflection in both the noun and the article. The student's intended answer plays a strong role in such circumstances, as it finds expression in not just one but several inflections.

Sometimes one word governs the other, as subject-verb; other times, both take their specification from somewhere else, as case and number in a noun phrase. In either situation, the common source (step 2 in the rule framework) should mean a common value for the indicated category, even if it is actually a wrong value. The student deserves credit for correct performance under the "internal agreement" aspect of step 2.

A three-way comparison may well be indicated. The individual words must be checked against each other, and the result of that comparision checked against the expectation. If there is any disagreement the expectation must be checked against each word individually (Perhaps also necessary to investigate possible weak-correct situations). One approach to the multiple-word problem lies in the pairwise application of the same comparison technique as for a single form, thereby reducing the problem to a manageable form. For two words, say an article and a noun, several distinct possibilities exist:

1. all three agree — fine, correct;
2. the inflectional characteristics associated with the actual forms agree with each other, but not with the values of the expected slot — a fairly clear instance of "internal agreement" on an incorrect value;
3. the characteristics associated with the actual forms do not agree with each other, however one of them agrees with the expected value — The latter form is correct, the other wrong, and there is no internal agreement.

Unfortunately, because of the multiple characteristics associated with a given form, the following confusing situation can arise (using a single letter to represent an arbitrary slot):

Expected slot: A
Word 1 - actual characteristics: X, Y (two slots)
Word 2 - actual characteristics: Y

In this instance, there is internal agreement for characteristics "Y", which is not in agreement with the expected value. At the same time, the expected value matches up well against the "X" characteristic of word 1, a characteristic not shared by word 2; a situation which could be interpreted as a form error in word 2. To take a concrete example,

Expect: "sie trinken" ("they drink")
Slot: <3 p *> (3rd person plural, any gender)

Actual: "sie trinkt" ("she drinks"?)

Forms used, and associated characteristics:

"sie" — <3 s f>, <3 p *>
"-t" — <3 s *>

Diagnostic analysis:

Expected - subject agreement on <3 p *>
Subject - verb (internal) agreement on <3 s f>
Overall: no overall agreement

The diagnostic picture becomes even more cloudy when the comparisons between slots are not exact, but must rather be referred to something like the closeness metric to determine which is less wrong. The symbolic example might then have X and X′, Y and Y′, and a judgment would be needed about how X matched Y′, compared with how Y matched Y′, compared with how X matched X′, compared with how X′ matched Y: instant insanity.

Some relief is afforded by the specifics of German, in that an multiple-inflection situation usually is quite degenerate in terms of what categories are definitely indicated by which words. Most often, the inflectional forms complement one another, adding precision to what would otherwise be ambiguous. Thus the "die" feminine/plural confusion is resolved by the noun ending. the noun shows nothing of gender in its ending, and the ambiguity of nominative/accusative for "die" is not resolved in either word′s inflection. The number of outright conflicts is thus less than might be imagined at an initial, theoretical investigation.

When Method 2 (distance metric) was being used for article inflection, I put it to work on the combined inflection problem, too. It was abandoned because it just didn′t seem to reflect the right balance between what one word showed alone and what effect it had on the other word (and also because of the difficulties of size and complexity mentioned above). For Method 1 (tabular) to be used in his context, some more information would go into the table, or the tabular results for the two words might be combined by some other algorithm, as yet unborn. I shall be happy if Method 3 (probabilities) allows the necessary interaction to be expressed in a clean formalism.

Subject-Verb Inflection Checking

Implementation of the verb-subject check has been reasonably satisfying. It provides a modest illustration of a number of the important ideas of the grading program, though circumstances have combined to make the demonstration far from complete in many instances. I have, so far, only implemented analysis of the present tense. In that tense, verbs have only five forms, showing only two basic grammatical categories: person and number. If the subject of the sentence is a noun or noun phrase, the available alternatives are further narrowed, as only 3rd person is then relevant. Pronoun subjects range over all values of person and number; but long before the verb analysis is begun, the pronoun search logic takes care of deciding which slots are actually pertinent.

The rule governing verb inflection demands agreement with the subject and the program has always concentrated on just that demand. If the subject is not what was expected, I still base the ending check on what is actually present. In terms of the general inflectional theory, the checking is defective, as it lacks a comparison with the expected value. Because so very few forms are involved (and one of them is identical to the infinitive), there didn't seem any point in going to a more involved calculation. In several ways, however, a check with the expectation does enter into the deliberations:

1. the determination of subject characteristics is done in consultation with what should occur;

2. those inflectional characteristics which depend on the verb type (regular verbs, irregular, etc.) are checked through a comparison with the correct type for that verb;

3. as the scope of the implementation is expanded, future versions will have to consider the expected tense and mood, particularly when dealing with irregular verbs.

I am intrigued by the two-way checking which sometimes occurs between subject and verb ending. The primary check is direct from subject to verb: find the subject, figure out what its grammatical characteristics are, and then check if the verb ending agrees. However, there may be some ambiguity in the grammatical attributes of the subject; the verb ending may be able to shed some light on the confusion. In particular, the verb inflection shows a very clear distinction between singular and plural; nouns and pronouns may be ambiguous in their indication of number. The pronoun search already contains logic to look at the verb ending when a subject pronoun is involved. The noun phrase logic does not yet make a similar check, but should. The two-way check is not without its problems, not the least of which are the 2nd person familar forms. But even in limited implementation, the verb two-way check provides a nice example of how an examination of both parts of a related inflectional scheme can aid the determination of just what the student had in mind.

To handle irregular verbs, it was necessary to expand the verb inflection logic to include a consideration of the stem as well as the ending, and to add another grammatical category to reflect the type of verb. (See above for a complete discussion of the "type" category.) A modest multiple-term inflectional situation thereby exists: subject, stem and ending. My basic strategy is to attempt to form a composite from the stem and ending which can represent the whole verb, and then compare that composite with the subject. To properly diagnose errors, individual comparisons of subject with stem or ending are often required.

The exact nature of the comparison varies slightly from one stage to the next. Throughout, the very simplest comparison scheme is employed: values match or they do not, with no half-correct measures considered. Of course, a particular inflectional form for stem or ending will have several character slots associated with it. In forming the subject-end composite, the program evaluates all possible combinations between the two sets of slots. Only those in which the values for both person and number match exactly are retained. Each slot includes the specification of one or more verb types: the intersection of the verb types is formed for the composite terms. In comparisons involving the subject, no verb type is relevant and the program only need check the exact match of person and number.

By checking subject against the stem/ending composite for an exact match, the program can tell directly whether the inflection is proper. If there is an error in the stem, the subject/stem comparison will bring that out; likewise for ending error. Another possible diagnosis is that of verb type error. The program must check the verb type of the final comparison term against the correct type for the verb. If there is no match, then the inflectional form would be correct except that the student has not properly observed changes due to verb type. Other type errors may be localized in the stem or ending, and can be easily spotted by a similar comparison of types.

A full description of the verb inflection check, along with several examples of actual program output, can be found in Appendix D.

Chapter Eight

WORD ORDER ANALYSIS

Introduction and Definition:

A major grammatical phenomenon, that known as "word order", is based on the order (sequence) of words in the sentence. In my work, I use an extended notion which also covers the order of phrases and elements, and includes the determination of the basic clause type. Like inflection, word order as a grammatical phenomenon appears in differing roles: in the identification of the individual words of the student's sentence; in deciding what the assembled element or sentence means; and as a number of grammatical rules which, in the instructional setting, must be checked for correctness by the grading program.

The general analysis used in this project considers the German language as having three grammatical levels: words, clause elements, and clauses. This tripartite division is of particular pertinence to word order, whose role centers around the two transitions: word to clause element, and clause element to clause.

From Words to Clause Element

A clause element is built out of words, basically, with perhaps a little internal structuring. The meaning and grammatical function of the element derives mainly from the choice of words, along with inflectional indicators. Word order within the element is not very interesting. The German language allows little or no flexibility of order; the rules are there to be followed, serving perhaps a useful function in allowing one to group together the consituent words of the element without being confused by neighboring ones. Accordingly, I use word order within the element as a major guide in the identification process, looking for modifier and functional words where they should be located relative to the head word of the phrase (see Chapter 5). Particularly for the simple structures which the program can accept at this time, I consider within-the-clause word order errors unlikely, and provide no diagnostic facilities for them.

For reference, the current multi-word element capability is as follows:

(noun phrase) : article + noun

subject or object : noun phrase [article + noun]

: preposition + noun or pronoun

: preposition + noun phrase [article + noun].

In the near future, a certain amount of flexibility will be required due to a planned expansion in the complexity of noun phrases. For instance, I will include phrases with two nouns, as in "eine Tasse Kaffee" ("a cup of coffee"). The search routines already have the needed flexibility; what is lacking is the error checking and interpretation.

From Clause Element to Clause

A clause is constructed out of clause elements. By careful design, the word order analysis of a clause is deferred until all the constituent words have been grouped together into elements. As remarked above (Chapter 3), word order (at the element level) is not used in searching for elements, neither in identifying them nor in determining grammatical function. This follows from my general philosophy concerning errors, that nothing which the student might do wrong should play an important role in identifying his response; also from the inherent flexiblity of the German language regarding the ordering of elements, which tends to preclude any prediction as to what element will come next (particularly among sentence field elements, where help is most needed).

Two Tasks: The task of the word order portion of the program is twofold. The program must check the order of the clause elements for conformity with rules of grammar and with the shade of meaning incorporated in the expected sentence. Furthermore, the program must determine, by analysis of the actual student response, what type of clause has been created (main or subordinate; statement or question), and give an interesting diagnostic report on its findings. The clause type determination is done first, as some parameters of element ordering depend on the type of clause.

Clause Type Determination:

The clause type determination subscribes, in theory, to the three-step model: cue, intent, actual response. However, there are only a few forms, far less than for inflection. Also, the nature of the verb placement rules is such as to leave little ambiguity as to what was intended. Most likely, the student will err either in a wrong intent or in partial distortion of the execution step; in either case, his intent will show quite clearly.

The first step, then, is to determine what he seems to have done. Then, if that is not correct, considerations of motivation will be in order. Errors in execution may confuse this initial determination. It may even become clear that the student does not know the applicable German word-order rules at all, in which case a diagnosis of "confused" will be returned.

For reference, here are the clause types currently included in the program's German grammar competence. For each type, I list the identifying characteristics:

TYPE	CHARACTERISTICS
Statement	FV-2: Finite verb in second position, i.e. at front of clause preceeded by exactly one clause-element.
Information Question	("Who is that?") FV-2: same verb position as statement. Initial element will be the question word.
Yes-no question	("Are you coming?") FV-1: Finite verb is at the very front of the clause.*
Dependent clause	FV-L: Finite verb is Last, at the end of the clause, following even a dependent infinitive or past participle. The initial element will be a relative or subordinating word.

(The "finite verb" is the inflected verb, as distinguished from an infinitive or past participle. The "FV-n" notation refers to the finite verb position, and is borrowed from Kufner <ref. 7.2>.)

As shown in the chart, verb position and introductory words are the main factors distinguishing different clause types. There are several other differences: the dependent clause FV-L configuration is the only one in which an infinitive can precede the finite

---

* Or occasionally it may come second; see <ref. 9.3; Reference Grammar, B786>.

verb; a separable prefix will not be detached from a finite verb in FV-L. The program also relies on its knowledge of the expected type, particularly:

1. when an error results in conflicting indications (i.e., an initial relative word, but no FV-L);

2. when a clause has so few elements that a clear determination is hard to make;

3. to decide between command and yes-no question (not yet implemented).

Type Determination Algorithm: The clause type determination algorithm begins by breaking the nonverbal elements of the clause into three groups: those before the verb, those between the two verbs, and those following the second verb. (If there is only one verb, the last category is null.) It can then make the following pattern tests:

("FV" denotes Finite Verb, "DI" Dependent Infinitive)

— no elements before FV means FV-1 format;

— no elements after FV probably means FV-L format, particularly in DI precedes it and/or there is a Rel. Word present. However, the simple form <subject + FV> is FV-2, a statement;

— one element before FV is a statement, FV-2.

To handle erroneous formations, the actual patterns are somewhat broader than the above, to wit:

— DI preceeding FV probably indicates a dependent clause (FV-L); confidence diminished if no Rel. Word, or if FV position is too far from end of clause;

— FV last position can indicate a dependent clause even if the explicit Relative Word is missing;

— FV last position can indicate a dependent clause even if the initial word is a Question Word (unless there is nothing else before the verb);

— FV in the middle of the phrase is tentatively identified as FV-2, particularly if a DI is present and follows the FV.

Various grammatical checks must be made:

— for FV-1 & FV-2, if a DI is present it must come at the very end of the clause; DI following immediately after FV is taken as a specific mistake, perhaps an indication of an English pattern;

— for FV-L, DI (if present) must be at the end of the clause but preceeding FV;

— for FV-2, FV must really be the second lement;

— separable prefixes associated with DI must always be attached;

—separable prefixes associated with FV must be detached except for FV-L.

The type determination is accompanied by a level of confidence, evolved dynamically during the course of the analysis. The initial guess will usually have "high" or "veryhigh" confidence. Errors will cause the program to successively lose confidence in its diagnosis; if a level "none" is reached, the determination will cease with a value "confused". Likewise, certain observations may boost the confidence level; correct performance in placing a separable prefix is a notable example.

A full description of the algorithm is given in appendix E, followed by examples of its operation on more than a dozen sentences in appendix F.

<u>Diagnostic Interpretation</u>: Interpretation of the clause-type findings requires additional analysis. The three-step model will be most useful: cue (expectation), intent, and execution (actual form).

The student must make two decisions in deciding on a clause type: first, whether it will be a main or subordinate clause; second, if a main clause, whether a statement or question form is required. I shall first consider the determination of type for a main clause.

The decision as to statement or question is based primarily on semantic criteria, what the sentence is to say. An error, then, is not so much a violation of a grammatical rule as it is the production of a different sentence. I do not consider such interchange error very likely, other than due to the student misunderstanding instructions. I find it hard to say whether an error would be due to improper intent or faulty execution, as the required execution is fairly straightforward and natural for an English-speaking student.

Specifically:

1. statement becomes FV-1 Yes-no question: probably intentional (no English pattern beginning a statement with a verb);

2. statement becomes FV-2 Information question, or vice-versa: probably intentional, as the use and meaning of the introductory question word is quite clear;

3. FV-1 Yes-no question becomes FV-2 statement: Possibly intent wrong, or possibly execution confusion on an English pattern ("Today is anyone lonely?").

The choice between main and subordinate clause is determined on a purely grammatical basis from the structure of the sentence: one main clause, all others subordinate (with the usual exceptions that prove the rule, particularly quotation). I anticipate many errors just at this point, if only because English allows one to use subordinate clauses without being aware of so doing and without the subordinating word. The clause type determination takes into account that the student may use some of the trappings of a dependent clause, particularly the subordinating word, without realizing (and thus without intending) the subordination. Or, conversely, he may consciously produce a dependent clause without a subordinating word.

There is no way to be certain whether the error is in intent or execution, though some guesses can be put forward on the basis of available data. For demonstation purposes, I have set up a decision network to handle the guessing. It is guided primarily by the two main markers of dependent clauses, the subordinating word and the finite verb-last position. The conclusions are as follows:

performance		: diagnostic interpretation	
Sub.-word presence	FV-L?	Intent	Execution
ok	ok	correct	correct
ok	no	maybe — or — maybe (either one right, other wrong)	
missing	ok	prob. ok	error, missing word
missing	no	wrong	must be referred to what was done instead

The above chart, which has been implemented, could be augmented by information from the placement of dependent infinitives ans separable prefixes. At this time, they are used the basic type determination but not in this secondary analysis.

Checking of Clause-Level Word Order:

Having decided on which type of clause is present, the analyzer is ready to deal with the element word order of that clause. All preliminary work is out of the way: the clause limits have been defined; all the constituent elements have been formed from the individual words and analyzed; any information needed from the actual performance, such as the determination of clause type, has been done; and, of course, there is the information on what is expected.

Clause word order is governed by various kinds of rules: some, like the verb placement rules encountered above, are very strict, closely prescribed, and of primarily grammatical import. Other positional decisions can be made more flexibly, with word order determined by the desired meaning or influenced by the particular emphasis desired.

Meaning determinations are connected with the need to establish grammatical identity for each element. For example, the distinction between subject and object is sometimes made on purely positional grounds. Or meaning may figure into word order because of the varying modification involved in different positions. The negative adverb "nicht", for example, has a different interpretation depending on where in the clause it occurs. Emphatic shifts occur as certain elements are placed in different order. This applies especially to the choice of the initial element, but also to other ordering considerations. (This kind of emphasis is familiar in English mainly as stressed words when speaking: "Hans will buy the book". versus "Hans will buy the book"; in German, word order is a common factor, as in the less common English example "Over the house the little airplane flew.")

The various word order rule checks are best described one category at a time.

Verb Placement: Verb position is very closely prescribed. The dependent infinitive (or participle) will usually be at the end of the clause, followed only by the finite verb if a relative clause, and an end field if there is one.* The finite verb must be either first, second or last depending on the clause type. (The program will accept a statement, i.e., FV-2, with a dependent infinitive in the front field.) Verb position is not only a rigidly specified item, it is central to the whole element word order of the clause. The program accepts certain

* I am dealing with a simplified grammar, of course.

distortions of the correct pattern as reasonable mistakes. If the student's performance is worse than these limits, I merely assume that he does not know German word order at all. The following account of the program's specific case handling does not include a category of "elsewhere". A verb position other than those listed will usually cause either a failure of the clause type determination, or a verdict of "no confidence" in that determination; in either event, further word order checking will be discontinued. (The following Algorithm is part of the PHRASEDIVIDER / PHRASETYPEREPORT complex; see appendix E for full description.)

Finite verb position for a statement:

position 1 —— not recognized as statement

position 2 —— correct

position 3 —— incorrect - "front field error" but tolerated. In the presence of this error, no checking is done on the front field (see below).

Finite verb position for a dependent clause:

at very end —— correct

followed by one element —— correct if that element can be an end field; otherwise, clause may not be recognized as dependent.* if so recognized, tolerated as incorrect, and offending word identified as an illegal end field.

---

* Recognized only if a dependent infinitive preceeds or if there is relative word and some other element before the verb.

Dependent infinitive position in main clause:

at very end — correct

followed by one element — correct if that element can be an end field; otherwise, incorrect but tolerated. The offending element is identified as an illegal end field.

neither, but immediately following finite verb — incorrect - "possible English pattern" - but tolerated. (Compare English "I can buy a new car today.")

Dependent infinitive position in a dependent clause:

following finite verb — incorrect - "finite verb must follow infinitive in dependent clause"

immediately preceeding finite verb — correct

elsewhere — incorrect - "random positioning" not tolerated but if finite verb position is okay processing will continue, ignoring the position of the infinitive.

Sentence Field: The "sentence field" elements include the subject, objects, modifiers, and many predicate complements; in short, just about everything except the verbs themselves. In a statement, exactly one of the sentence field elements is placed in front of the finite verb, in the "front field".

A brief presentation of the relevant definitions and rules of German grammar is available in appendix A.

Clause word order is mainly a question of the ordering of the sentence field elements and the selection of one element for the front field. As the front field element, if it even exists, is simply removed from the sentence field, the program is designed to handle word order for the full sentence field. If an element is removed to the front field, its absence is easily noted; if the student selects a different front field element than the expected one, the word order analysis can still proceed directly.

Word order within the sentence field tends to be specified in rather vague terms, even by the Reference Grammar; various rules are offered to the students, but a certain reliance is placed on what sounds right. The program can be given the expected word order for the sentence and compare that against what it receives from the student. Not only would I like the program to be more self-sufficient; a greater internal knowledge is essential to be able to produce reasonable error messages.

The most important ordering rule is that called "news value." When this is invoked over other conventions, the program is totally at the mercy of the expectation information. The other conventions available to guide the program are assembled in a "normal word order". Where possible, I have had the program check adherence to the normal word order rules:

1. pronoun elements (subject or object) must preceed noun elements in the inner field. They cannot be shifted to the end of that field for added stress;

2. pronoun subject must preceed a pronoun object;

3. noun-phrase object with an indefinite article follows one with a definite article. (Not yet implemented; given in text only through examples, not as explicit rule).

I am not certain how best to approach the rest of the checking. The following considerations apply:

1. A way is needed to compare two orderings, such as the expected with the student's. Simply counting the number of permutations is probably not optimal from an educational viewpoint; although some errors are simply due to confusion, others are doubtless more systematic and should be so explained. I have done some preliminary investigation into an appropriate algorithm, but not enough to report here.

2. Since there does exist a normal word order, the student's performance will have to be compared with that as well as with the expectation in order to catch errors of omission as well as comission. That is, the expected may differ from normal in one way, but the student produces a sentence without that deviation, rather some other. Without reference to the norm, the program would be forced to diagnose an interchange of elements, rather than a "you did this instead of that" report. For

example, let normal be "A B C D E", expected be "A B D E C" and the student's clause stand "A C D E B", assuming (as is the case) that there is an applicable rule governing the moving of an element to the end. The student, clearly, moved "B" instead of "C", but a pure comparison of his with the expectation will say that "B" and "C" are exchanged.

Sentence field word order may have direct grammatical role under some circumstances. Ordinarily, inflectional changes are sufficient to distinguish between subject and object; or the semantics are sufficiently unambiguous (e.g. houses do not buy people). If neither indication suffices, then the sentence must adhere to normal word order in order to be unambiguous; that is, the subject preceding the object. (This check is not yet implemented.)

Front Field Choice: If the clause is to be a statement, it must have one element preceding the finite verb, in the "front field" slot. This element is taken from the sentence field, as noted above. If the front field contains more than one element, the program will declare a verb position error, as noted in the chart above. Since the student is obviously confused about the defintion of the front field in that case, no analysis is done on its constituents. If the front field is correctly formed, the program must decide by reference to the expected sentence whether the correct element has been chosen. The choice is purely semantic, so there is little the program can say other than to report correct or incorrect. (The size check is currently implemented, the choice of element not.)

<u>Front Field or End Field Elements</u>: If a clause element consists of a dependent clause, it must be placed in either the front or end field. (The "end field" follows everything, even dependent infinitive.) The program can easily check this rule, as well as referring to the expectation information for what might have been done. (Placement check is implemented; reference to what was expected, not yet.)

<u>Separable Prefix</u>: The separable prefix which characterizes many German verbs falls under rather definite positioning rules: it is separated from a finite verb in a FV-1 or FV-2 clause, otherwise it must be attached. The program currently goes to a lot of trouble on account of these prefixes, partly because it checks the disposition of the prefix as part of the clause type determination, even though it is not until that determination has been made that the prefix check can be done properly. If the prefix location is inconsistent with the clause type as decided, the confidence level in the type determination is reduced.

In reporting the prefix position, the program distinguishes infinitives, for which the prefix must never separate, from finite verbs. Four categories of placement are detected: attached; detached but immediately preceding the verb, which is treated as attached with an orthographic problem; detached and at the end of the clause; and detached in some random location. The attached and detached-(proper location) cases are then compared with what is proper and messages are generated accordingly.

The detached-(random position) case should receive more attention, as it may actually indicate a different confusion. The separable prefixes are homographic with prepositions. Since many verbs take prepositional objects, the student may indeed have a valid confusion.

Introductory Words: Question words and subordinating words must be found at the very beginning of the clause. There is a possibility of confusion from an English pattern which allows a preceding adverb:

"Who's on first now?"

"Now what'll we do?"

as well as the less common, contrastive:

"He went where?"

The program takes some cognizance of these error possibilities; it can analyze a question clause even though the question word is not first, and complain later about the position error.

"Nicht" (not yet implemented): The adverb of negation, "nicht", presents some special positional problems. Placed at the end of the inner field, it negates the whole clause; placed elsewhere, it negates only one element or conveys a different emphasis. The program will have to check for legal positions as well as explain any variation from the expected position.

Negative sentences pose another, related problem, particularly when only one element is to be negated. The student may use a negative article "kein-" instead of "nicht"; or, more pertinent to word ordering, he may place the element to be negated in the front field, complicating the positioning problems for "nicht".

Particles: German has a class of words known as particles or intensifiers. These do not count as clause elements, and can appear almost anywhere between elements. A simple check with the expected sentence is about all that can be done here. (The current implementation does not check what was expected.)

PART FOUR

Conclusion

and

Recommendations for Future Work

Chapter Nine

CONCLUSION

This dissertation originated in a desire to fashion the computer into a tool for performing sophisticated answer analysis, i.e., grading of student responses. I have provided a working program which, while still modest, does represent a successful demonstration of my principal ideas.

In the course of the research, I have developed algorithms to perform many different tasks. Some of these algorithms are noteworthy in their own right. Of equal importance from a technical viewpoint, I believe, is the assemblage of many routines into an integrated whole. The interrelationship of the various parts allows the program as a whole to surpass its individual components and, in so doing, provide an indication of the more refined performance which will be available simply by upgrading the various constituent routines.

As another achievement, I claim a good beginning of a sound theoretical base for the analysis and explication of errors. My theory provides guidelines pertaining both to the process of pinpointing an error and identifying its probable cause.

In all honesty, several aspects of the current implementation represent but a modest success. For one thing, the program can presently accept only a very limited range of grammar. However, the limitation is due largely to considerations of time and effort on the part of the researcher; I see no major obstacle in the path of increasing the grammatical complexity significantly while retaining the program's present structure.

The program lacks a formal grammar, making assertions about the state of completeness of the grammatical checking very difficult. Yet I would venture that a great many of the interesting features of natural language may be most profitably considered as a collection of special cases; if this be so, a flexible approach, capable of easy accomodation of any special case checking, is indeed appropriate. The issue of completeness can be settled by the Reference Grammar, by simply insuring that checks have been programmed for all the appropriate rules.

A serious shortcoming lies in the limitation of the program to a single, specified sentence. This difficulty is rooted in the basic design of the analyzer; even so, there are several ways in which the program can be improved to soften the impact of the limitation. For specific educational applications, the limitation is not always a terribly unnatural one. In sum, this difficulty represents but one of many aspects of the program; it is not an overriding failing, but must be weighed in the balance with the other positive and negative features.

The text of this dissertation has detailed many positive contributions. I might list a few here in summary: the ability to analyze entire sentences, i.e., complex structured responses; the production of detailed error reports; the ability to cope with multiple independent errors; the assignment of proper credit to each aspect of performance even in the presence of related errors. Most important, I have aimed at construction of a grader which would not be a liability to an instructional program, but rather would be able to lead the way to more imaginative teaching through its ability to gather large quantities of information on what the student has done. I am particularly proud of the program's ability to simultaneously follow diverse aspects of the student's general performance.

Chapter Ten

RECOMMENDATIONS FOR FUTURE WORK

There is no shortage of possibilities for future development of this project. The capabilities of the analyzer can clearly be increased markedly on a number of fronts. The educational promise of the grader can be realized by constructing an instructional program to utilize the information extracted by the grader. And on a broader scale, many of the general principles I have elucidated, as well as some of the specific techniques, could be applied to Computer-Assisted Instruction in other subject areas.

Extensions of the Existing Algorithms

The program has many loose ends. Some, like the restriction to present tense verbs, are the result of simple constraints of time and effort in any particular area. In other instances, such as word-order checking, the incomplete state of the grammatical analysis also reflects my need to have something to show for the overall demonstration of the grader. In the latter case, upgrading the program's capabilities will probably entail some local rebuilding.

Overall, the program is in fairly good shape concerning extendability, as some of the more recent additions testify. The addition of facilities to handle prepositions, for instance, was quite straightforward and simple. A characteristic function was created to determine a word's membership in the set of prepositions, implemented trivially using the LISP property-list feature. A simple function, modeled on those used for noun phrases, established a structural context for the analysis of the prepositional phrase; the mechanics of scanning back from the phrase to claim the preposition was already contained within the general executive routines.

In another instance, I once decided to have the program automatically adapt to the paraphrase of a noun phrase by an equivalent personal pronoun. At the point where the program noted the absence of the head noun, all of the information was already available to specify the pronoun, relying on the program's internal knowledge to determine the orthographic form as well. There was no difficulty fabricating the necessary search command, by duplicating the equivalent Phase One activity; the mechanism for carrying out the search is precisely the routine already used for pronouns.*

* The paraphrase feature was removed during a subsequent reorganization. It is easily reintroduced as a special case, but I have preferred to delay until I could make it a part of a more comprehensive check on extra and missing words.

Here are some of the ways in which the program might be improved.

Additions to the Grammatical Capabilities:

There is no limit close at hand on the expansion of the program's grammatical routines, even restricting consideration to the needs of a first or second year language course. The improvements suggested here represent those that seem the most immediate candidates at this time.

Verb tense expansion, especially to past and perfect tense: This requires that the program have the ability to distinguish the conjugational patterns of verbs in the past tense, not too great an extension from the current state. The formation of past participles is more complex, due to interference from prefixes and the preposition "zu". Furthermore, the dictionary information associated with each verb would need to specify which auxiliary is required to form the perfect tense. More complex checks are required to automatically detect the substitution of one tense for another, particularly with the perfect where an extra word is involved. (Note that the verb analysis is already designed to delay its decision whether a particular verb is indeed the finite verb of the clause, precisely to handle the case of an unexpected (or missing) auxiliary.) A rather significant effort is required to assign a semantic interpretation to a interchange of tense once it is detected.

Articles and Pronouns: Some of my more recent ideas in error theory need to be applied to the pronoun analysis, particular those dealing with the representation of strong and weak correct. Similarly, the probability method for checking article inflections should be implemented and tested. More mundane topics include article substitution analysis, which needs considerable refinement, including some semantic consideration for certain cases and pronoun forms relating to the second person familiar, which must be handled better.

Negation: The program should be able to handle negative sentences, preferably through an automatic mechanism. It should have internal knowledge of how to negate an arbitrary sentence, as well as diagnostic capability. This will involve position criteria for the adverb "nicht" and rules relating to the use of the negative article "kein-".

Word order: The check on the order of the sentence field elements must be implemented, comparing against both normal word order and the expected ordering. I have addressed this topic to a certain extent in the main discussion of word order (Chapter 8). The problem is very difficult, involving a number of loosely formulated rules and appeals to semantic criteria; it may well require that additional information be provided as part of the sentence description.

Inflected Adjectives: These will provide a profitable area for further work in complex error analysis. The programming will involving attempts to match up the inflectional patterns shown by the adjective with those of the noun and article, and all that against both what is expected and what is grammatically correct.

Additions to the General Analytic Capabilities:

Detection and recovery from misspellings constitute an essential part of further development. I am aware of work on misspellings in English <ref. 18; Symonds, 1970> and French <ref. 14.3; Scholl, 1972> involving consonant substitution and vowel elimination as part of the matching algorithm. Quite possibly similar efforts exist for German. I think that a moderate approach would probably suffice for this project, to catch such difficulties as the "sch" phoneme, or the "ig"-"ich" suffix interchange. Clearly some discussion with experienced German teachers is indicated. In terms of the program, however, I insist on attempting a maximum separation between inflectional changes and misspellings; I will not have the misspelling routines interfering with the grammatical analysis, nor a grammar error detected only as a misspelling.

Logic to deal with the capitalization of nouns and some pronouns would represent a good addition. Some convention is needed for conveying the upper and lower case distinction on a teletype, of course; the search functions would need modification so as not to be distracted by the case-shift indicator. As far as checking is concerned, all the word analysis routines already call on a common termination function; that function could easily be expanded to do the necessary checking.

Curriculum recommendations stemming directly from the analytic process should be made available. To clarify this notion, consider that the major portion of a grading analysis report concerns what the student knows and does not know, information of direct interest to a teacher who must decide which curriculum to present. There is another area in which the operation of the grader is pedagogically interesting, to wit, the limitations which two obstacles impose on the grader's ability to gather information: the inherent ambiguities in German grammar and the shortcomings of the program's guidelines for the interpretation of errors.

The program is uniquely able to detect when it is in trouble; what it does with that information is another matter, clearly indicating an area for future research. Currently, the weak correct diagnosis is the major current manifestation that the program has encountered a grammatical ambiguity; but not all of the inflectional checks (and none of the word order checks) correctly detect weak correct situations. In addition to merely reporting that a weak discrimination occurs, the program ought to be able to point out how a different formulation of the expected sentence would have avoided the particular indeterminacy.

Consider a simple example, taken from inflection checking. The student is to produce a dative neuter article, i.e., "dem", but instead has written "das". His mistake can be interpreted in two ways, either as an accusative neuter (not a bad guess if dative is correct) or a nominative/caseless neuter. As both have the same form, there is no way for the program to be at all sure what the student had in mind. However, if the sentence had been such that a masculine article was required, the particular inability to distinguish nominative from accusative would not occur. For the program to make a comment to that effect, such as "try a masculine noun in order to clear up this particular ambiguity," should not require a very great addition to its grammatical capabilities.

In the discussions of various techniques of error interpretation for inflection, I described an algorithm which would have the ability to figure out the appropriate curricular recommendations to get around grammatical ambiguities.* Implementation of that algorithm would seem a good place to begin. For word order, I have not yet really addressed the problem of reporting analytic ambiguity

---

* See Chapter 7, description of probability method.

and the ways to avoid it. The level of confidence given by the PHRASEDIVIDER package reflects analytic troubles, but those due to grammatical ambiguity are not separated from general confusion by the student. I have a few general ideas of how to proceed with the necessary work; I am confident, again, that the additional capability will represent a fairly natural extension of the existing program.

Additions to Increase the Program's General Awareness:

The search functions should pay more attention to context. For instance, they should consider whether a preposition been seen, or if an expected combination of adjective and noun is present.* To achieve such increased attention will require more elaborate preprocessing operations during Phase One of the analysis. For the program to be aware of the various contextual considerations which will be needed, a number of data structures will have to be set up for a specific sentence. Furthermore, each of the search routines will need modification to provide for handling additional parameters, and to introduce a certain amount of intercommunication between the individual search functions.

* See: discussion of limitations in the word match algorithm, Chapter 3.

Dependent clause analysis is currently hampered by a certain vagueness in the setting of clause boundaries. Some of the difficulty stems from the deferral of the pronoun search until the very end of the search. During the regular scanning of a clause, a correct instance of an expected pronoun should be immediately claimed by the appropriate function, just as is done for nouns and other major words. In any event, even when deferral is necessary, the program should be generally aware of the existence of unclaimed pronouns in the interior of the clause, so that it can tell whether to proceed past the last major word of a clause during the pronoun search. Note that if the student's syntax is correct, there is no problem determining the clause boundary; the dependent clause will be delimited on the front by the subordinating word and at the end by the finite verb. But the student may not be correct, in which case a pronoun may be the last word of the clause he creates.

A major area for additional work concerns the analyzer's response to word presence errors: omission of expected words, addition of unexpected extra words, or substitution of one word for another. This topic is a Pandora's Box, as it quickly turns into a question of analysis of arbitrary natural language, a problem I'd rather stay away from at this point. Nonetheless, there is quite a bit of room for improvement even within the general constraints of the present research.

Pure omission is probably the easiest of the three to handle, as there is no need to cope with unknown words and their structures; yet a meaningful interpretation of the omision may be somewhat tricky, particularly if what the student has produced does have an interpretation in correct German. Pure addition of words, on the other hand, most quickly becomes unmanageable. Given access to a dictionary, the program might be able to identify some unanticipated words and deal with them using a small set of heuristics.

Substitution is perhaps the most interesting of the three types of presence error. It is strongly related to both omission and addition. When a substitution occurs, the expected word will not appear, an omission; an extra one will be present instead, an addition. If there is any logic behind the substitution, the information about the expected word will still be partly applicable; e.g., noun, head word of an accusative object, etc. Pronoun paraphrase is probably best treated as a special case of substitution.

When changes from the expected occur in the actual lexical words the student employs, the program must be prepared to respond with some diagnostic comment on the resulting semantic implications. This again is a wide open subject. Two observations, however. First, if the program can identify the unexpected word, using a dictionary, then it can perform the normal inflectional checks relating to that word even without knowing the meaning.* Secondly, the task at hand has been defined as the teaching of German grammar, and detailed discussion of finer semantic points might well be deferred to another time.

The program currently detects missing words in one of two ways. If a major word is missing, its WDLIST entry will remain even when the end of the clause has been reached. The absence of function and other words which are the subject of directed search will be noted at the time of that search. The absence of major words is currently noted in a most cursory manner; an obvious expansion would be to follow the cross-reference pointer to the corresponding SDLIST entry so the program could at least refer to the entire clause in making its complaint. That point would also be the time to consider whether a substitution may have occurred; most likely the program will have to maintain a list of unrecognized words from the main scan for reference at this point.

All the SDLIST functions, at least those corresponding to top-level clause elements, would have to be made sensitive to the prospect of being activated on a major-word missing basis, as well as the current major-word found condition. The case of pronoun paraphrase is an example of the kind of complexity that must be handled: the

---

* The inevitable special case, in which either the new or old word has some grammatical anomaly, should be noted in that word's dictionary entry.

SDLIST element would be activated first for the missing noun. Later, if the pronoun were found, it would be entered again as part of the pronoun analysis. if the pronoun were not present, then action appropriate to the omission of the original noun would be necessary. Other substitution analyses will be equally, if not more, complex.

For those words whose omission is noted during a directed search, a different strategy is needed. There is, of course, no problem detecting the absence. But because part of the clause element will be present in this case, the error interpretation may well be more involved than merely stating that such and such is absent. For instance, if both the article and adjective are missing from a noun phrase, that absence should probably be treated as a single event, not as two isolated errors. The solution seems to require that the [SDLIST] analytic functions do more than just activate lower level analyzers; when the lower level work is done, the calling function must delve into the resulting data structure and determine what has taken place, assign an interpretation to it, and so forth. For example, the Noun Phrase function currently just sets up a context for inflectional checking. Under the proposed improvement, this function would have the responsibility to notice other features pertaining to the noun phrase as a whole. For instance, if an expected article is not present, the noun phrase function would decide why the article was needed in the first place and produce an appropriate comment relating to its absence.

The same functions would appear to be the appropriate location for heuristics to handle detection and analysis of extra words. The middle of a noun phrase is a good legal place for an adjective, for example, and the Noun Phrase function should do its best to accommodate the student who insists on doing more than he is told. Perhaps the program should be sensitive to certain probable errors, much as this runs against my general philosophy of not looking for specific anticipated errors in the grading process. Thus in a construction in which one noun modifies another, the English pattern may gain the upper hand: "Bringen Sie mir ein Glas von Bier" ("Bring me a glass made out of beer"!) <ref. 7.3; Kufner>. An extra preposition might well be anticipated in this and several other environments.

Creation of an Instructional Program

The grader was designed for use by an enlightened computer tutor. Although the project is still very limited in scope, the essential characteristics are now fairly clear, enough to allow research to begin on an instructional system which would use the grader's special analytic capabilities. I am not entirely sure what form such a system would take. One of the design criteria was to provide a grader which would be an asset to a variety of instructional systems, rather than one rigidly linked to a specific instructional philosophy.

I see the [computer] teacher as being quite responsive to the individual student. A basic prerequisite of individualization is that the teacher have specific knowledge about the student; probably the more the individualization, the greater the amount of information needed. My grader handles large responses not just because they are pedagogically desirable, but also specifically to speed up the flow of information from the student to the teacher. In this context, the vast amount of diagnostic information takes on a new usefulness; rather than representing absolute verdicts for immediate (or nearly immediate) action by the teacher, the grader reports may just produce minor adjustments of the various parameters in a model the teacher might maintain to reflect the student's understanding. Curricular attention could be directed to a difficulty as "necessary" (an arbitrarily complex condition): perhaps after a series of small errors, or a pattern of scattered errors with one final straw clinching a hypothesis. Clearly a great deal of developmental effort will precede the appearance of an operational instructor. I would hope that the resultant program would have sufficient diversity and flexibility to pose an answer to the following eloquent criticism of mechanized instruction:

> The skilled teacher has many objectives to work from. She is not concerned about getting them all done ...[she can] seize the opportunity to do partial objectives well. They will reassign goals on the spot. The reconsideration of priorities is the important purchase that we make when we chose skilled teachers rather than programmed material. <ref. 16; Stake, 1967>

Applications in Other Areas

Language:

Close to home, the analytic techniques of this program should be applicable to the grading of other natural languages. As noted in the introduction, this project is very intimately involved with the specifics of the German language. Nevertheless, many of the general principles used by the grader have wider applicability. Special consideration will be necessary, of course, to handle the particular features of each language; but much of the underlying architecture may be useable. Thus, for instance, French articles must agree with the noun in number and gender. Although forms and patterns are different from German, the general notion of guessing what the stduent had in mind is still valid. Specifically, one of three methods (see Chapter 7) would probably be directly applicable, albeit with a different set of parameters.

Other Subjects:

In other subject domains, the grammatical analysis capability will be less pertinent than the general philosophy of my approach to grading. Language teaching has no monopoly on the notion that grading should be done via a program that is competent in the subject domain, that consideration must be given to questions of partial credit and multiple errors, and that reasonable explanations may exist to account for the students errors.

For a simple example, consider a problem that arises in elementary music theory courses: a short series of notes is played for the student, who must write them down. The student is not expected to have absolute pitch, only relative pitch; the object of the exercise is to train his recognition of intervals and transitions between notes. Therefore I would expect a grader program to have at least a minimal ability to transform a response to a different key. (For instance, if the sequence "C G D" were presented, the grader must be able to recognize as equivalent "A E B", "G C# A" and many others.)

To continue into the problem more deeply, consider the following difficulty. For simplicity, I will assume for the moment that the student's first note will always be considered as correct (and use it to indicate to the grader the required transposition of the whole response). The student may specify the second note incorrectly, but get the third one right. Clearly, a theory of how he is going about his task is needed. If he is merely recording the interval between successive notes, then he has made two errors: first to second, and second to third. Alternatively, if he is also functioning in some global context in which individual notes are related to all that comes before, then perhaps he has committed but a single error: the improper specification of the second note. Notice the associated questions of whether a single error should provoke a transposition of the remainder of the response, and indeed whether there might be an error localized to the first note which would also affect the particular transposition performed.

Continuing the analysis a bit further, one might consider the conceptual underpinnings of the problem. Perhaps the sequence of notes is somehow related to specific harmonic intervals. In that case, it would be worthwhile to check whether a wrong response could be accounted for by the hypothesis that the student has chosen a different but still appropriate interval. Other errors might be traced to the student not knowing the correct formulation of a particular interval.

In sum, clearly competence in the subject domain is necessary but not sufficient. A comprehensive approach to grading requires two other components: attention to various aspects of error analysis and a way of integrating information about the expected response with the ability to analyze the material standing alone. I have provided a working example of a grader for German grammar instruction, and hope that other projects will come to apply what I have learned for their own subject areas.

APPENDICES

Appendix A: Standard German Grammar, Briefly

Appendix B: Table for Inflection Method 1 (Tabular)

Appendix C: Table for Inflection Method 3 (Probability)

Appendix D: Verb Ending Check

Appendix E: PHRASEDIVIDER Flow

Appendix F: Phrase Division Test Runs

Appendix A

A Brief Presentation of Some Portions of Standard German Grammar, with Particular Import for Word Order Analysis

The Reference Grammar of the German Language <ref. 9.4; Lederer-Schulz-Griesbach> gives a quite straightforward discussion of the basic structure of a sentence:

> The PREDICATE is the most essential functional unit of a sentence, and includes all verbal forms of the sentence as well as many verbal complements. It ... forms the SENTENCE CORE.
> <ref. 9.5; sec. H110>
>
> Sentences ... consist of one or more CLAUSES, each of which contains SENTENCE UNITS (words or phases) which can be recognised as constituent members of the sentence. These sentence units are placed into a grammatical and logical relationship to one another by means of a predicate... "
> <ref. 9.6; sec. H002>

The German "predicate" consists of the verb and certain closely associated complements. All other parts of the sentence, including objects, (which in English are usually part of the predicate) the subject, and various modifiers are considered under the classification of "sentence units". They are grouped together into something called the "Sentence Field", which is most strongly identified as a description of physical location within the clause. The sentence field is delimited on both ends by the split predicate (whose parts are labeled V1 and V2), as shown:

Anterior limit   Posterior limit

| sentence field |

V1 .... V2

"Hast   du   das Buch   gelesen"

<from ref. 9.7; sec. H022>

The full declarative sentence has two other slots, an end field following the second part of the predicate, and a front or prefield before the first part. The end field contains one sentence unit which must be one of certain special types. The pre-field is usually assigned excessive importance by non-Germans because of our strong association of the position before the verb with the grammatical subject of the sentence. In German, the pre-field has very little syntactic importance; it does not even exist in many types of clauses. When it is present, its function is mainly semantic (a link to what has come before); syntactically, it is treated as a trivial positional transformation:

> DECLARATIVE STATMENTS normally have one sentence unit preceding the first part of the predicate. This unit in the prefield of the sentence fulfills the function of a contact

member with the preceding sentence of a connected utterance or discourse, or serves to arouse interest by referring to a particular concept. It neither loses its grammatical and logical function within the sentence, nor its claim to its normal position within the sentence field, to which it reverts if the sentence assumes a different form.

<ref. 9.8; sec. H051>

The movement of V1 to the position following V2 (the famous verb-last order of relative clauses) is similarly treated as a simple positional transformation with no further significance on the sentence structure.*

I have one change in the Lederer-Schulz-Griesbach scheme, in their particular distinction between predicate complement and predicate modifier. I will not deny that the notion of predicate complement is valid for the German language, nor the right of the teacher to present this concept as he chooses. For the grading purposes of the program, however, I would observe:

1. The distinction is subtle, and thus one may expect that a student will fail to make it properly.

2. Because the distinction is subtle, it is easy to add it as a special extra consideration for checking clause-element positioning. (Even the Reference Grammar

*. This is not quite accurate, as there is sometimes interaction between V1 and parts of V2, but that is a minor consideration entirely internal to the predicate.

tends to consider predicate complements along with sentence elements when discussing word order <ref. 9.9; sec. H233>.)

Some complements are definitely tied to the verb, and appear with it in a dictionary: the "separable prefixes" are a good example. They are further distinguished in that they cannot appear in the prefield; the example s20a is not possible in correct German:

s20.	Der Zug faengt an.	(The train starts up)
s20a.	* An faengt der Zug.	
	("*" marks an incorrect sentence)	

These complements clearly do belong with V2, as they are quite closely tied to it structurally. A large number of the predicate complements, however, are less strongly attached.

s21.	Hans wohnt in Berlin.	(Hans resides in Berlin.)
s21a.	* Hans wohnt.	(Hans resides..)
s22.	Hans arbeitet in Berlin.	(Hans works in Berlin.)
s22a.	Hans arbeitet.	(Hans works.)
s23.	In Berlin wohnt er.	(He resides in Berlin.)
s24.	In Berlin arbeitet er.	(He works in Berlin.)

S21a is not a complete clause, and so the prepositional phrase "in Berlin" is formally considered a predicate complement. S22a, on the other hand, is complete, and so the same phrase "in Berlin" is considered to add "non-essential" information and is considered a predicate modifier, which is part of the sentence field. Both s23 and s24 are grammatically correct, however. (s24 may be a bit unusual, but it is grammatical.) Following Kufner <ref. 7.4>, I would like to use an extended notion to apply to both sentence elements and loosely-bound predicate complements.

The particular predicate complements which I want to classify as clause elements can easily be confused with the modifiers. The Reference Grammar actually admits the narrowness of the dividing line, and offers as a test the criterion of necessity (as discussed above, s21 vs s22) <ref. 9.10; sec. H233>. The predicate complement is by no means the only required element in the sentence; the primary object is equally indispensible:

s25:	Ich kaufe ein Auto.	(I buy a car.)
s25a:	* Ich kaufe.	(I buy..)
s26:	Ich bitte um das Brot.	(I ask for some bread.)
s26a:	* Ich bitte.	(I ask ..)

S25a and s26a are not complete clauses, because they lack objects. Since an object may be a prepositional phrase in German, the distinguished position of the predicate complement becomes further eroded. Compare the three examples given so far:

s21:	Hans wohnt in Berlin.	(Hans resides in Berlin.)
s22:	Hans arbeitet in Berlin.	(Hans works in Berlin.)
s26:	Ich bitte um das Brot.	(I ask for some bread.)

Based on the above discussion, I claim that the distinction of predicate complement is not a major structural one. The distinguishing features are tenuous even for a formal theory; expect a clear distinction from students in an elementary course is unrealistic. I feel that incorporating a complexity which is not reflected in the behavior of the student represents a hindrance to the error-oriented analysis.

I have, therefore, adopted a simpler scheme than that of the Reference Grammar. I use Kufner's term, "clause element", which he defines as any constituent that can occupy the prefield <ref. 7.5>. In particular, clause elements include all the Reference Grammar's sentence elements, as well as the less closely bound predicate complement. I make up for the change by insisting that the distinction between predicate complement and predicate modifier be kept for determining the proper word order.

Appendix B

Table Used in Inflectional Analysis for Intent-guessing Method 1

Input: Article ending + noun inflection + expected value.
Values: Intent-guess, with confidence level.

\+ + + + + +

Description:

Data which will come from the student's response:

actual article ending — "ending"

noun stem/ending — "noun:" + "s" or "p"

"s" - (singular) no change from stem form
"p" - stem and/or ending changes, which:
"p (-n)" - include '-n' ending
"p other" - no '-n' ending

Data which will come from descriptive information supplied to the program:

proper case — "expected case"

proper gender — (just indicated as "m", "n", "f"; or "*" if no discrimination is needed

proper number — not considered in this table

Data inherent in the table:

From grammar and past performance:

slots for which the ending is correct

— given following the ending. Format:

< case number gender >

"*" is used when any value will do, as in plural where there is no gender discrimination.

a list is used if more than one value is appropriate, as "(nom acc)" for either nominative or accusative.

combinations which the student may have intended; includes all correct slots, plus plausible additions to accomodate likely errors:

— "possibilities" for each case heading

*** probability/confidence that a given combination does indeed represent the student's intent, as shown by his performance

— entries in the chart, given as:
"veryhigh"
"high"
"med" (medium)
"low"
blank for "none"

\+ + + + + +

Actual Table:

On following pages, arranged by ending.

ending: -er as in "der", "dieser", "einer"

correct for: <nom s m> <dat s f>
("ein-" forms only correct for <dat s f>;
included here, will be checked separately)

Expected case: Nominative

Possibilities:		<nom s m>	<nom p *>
noun: s			
	m	veryhigh	
	n	high	
	f	high	
noun: p			
	m	med	med
	n	med	med
	f	med	med

Expected case: Accusative

Possibilities:		<nom s m>	<acc s m>	<dat s f>	<dat p *>
noun: s					
	m	med	high	low	
	n	low	med	low	
	f		med	high	
noun: p					
	m		med	low	med
	n		med	low	med
	f		low	med	med

Expected case: Dative

Possibilities:		<nom s m>	<acc s m>	<dat s f>	<dat p *>
noun: s					
	m		med	high	
	n		low	high	
	f			veryhigh	low
noun: p (-n)					
	m		low	low	med
	n			low	med
	f			med	high

noun: p (other)
use accusative p entries

Comment: The <acc s m> category probably should have been <(nom acc) s m>, to indicate blurring of the case distinction.

ending: -es as in "das", "dieses"

correct for: <(nom acc) s n>

Expected case: Nominative

possibilites:		<(nom acc) s n>	<(nom acc) p *>
noun: s			
	m	high	
	n	veryhigh	
	f	high	
noun: p			
	m	med	med
	n	med	med
	f	med	med

Expected case: Accusative

noun: s
use nominative entries comm

noun: p
use nominative entries

Expected case: Dative

noun: s
use accusative entries, with diminished confidence

noun: p
use accusative entries, with diminished confidence

ending: -e   as in "die", "eine"

correct for: <(nom acc) s f> <(nom acc) p *>

Expected case: Nominative

possibilites: <(acc nom) s f> <(acc nom) p *>

noun: s			
	m	high	med
	n	high	med
	f	veryhigh	med
noun: p			
	m	low	veryhigh
	n	low	veryhigh
	f	med	high

expected case: Accusative

noun: s
use nominative entries

noun: p
use nominative entries

Expected case: Dative

noun: s
use accusative entries, with diminished confidence

noun: p
use accusative entries, with diminished confidence

Comment: built-in bias for plural will miss those nouns which do not have a special plural inflection.

ending: -en as in "den", "einen"

correct for: <acc s m> <dat p *>

Expected case: Nominative
"confused"

Expected case: Accusative

possibilities:	<acc s m>	<dat p *>	<dat s (m n)>	<acc p *>
noun: s	veryhigh			
m	high	low	low	
n	high	low		
f				
noun: p (-n)				
m1	high	med		
m2	veryhigh	low		
n	low	high		
f	low	high		
noun: p (other)				
m		med		med
n		med		med
f		med		med

Expected case: Dative

possibilities:	<acc s m>	<dat p *>	<dat s (m n)>	<acc p *>
noun: s				
m	high	low	med	
n	med	low	med	
f	med	med	low	
noun: p (-n)				
m1	low	veryhigh		
m2	med	high		
n		veryhigh		
f		veryhigh		
noun: p (other)				
m		med		low
n		high		low
f		high		low

ending: -em as in "dem", "einem"
correct for: <dat s (m n)>

Expected case: Nominative
"confused"

Expected case: Accusative
possibilities: <dat s (m n)> <acc s m>

		<dat s (m n)>	<acc s m>
noun: s			
	m	high	med
	n	high	
	f	med	

noun: p (-n)
use dative entries

noun: p (other)
use dative entries

Expected case: dative
possibilities: <dat s (m n)> <dat p *>

		<dat s (m n)>	<dat p *>
noun: s			
	m	veryhigh	
	n	veryhigh	
	f	high	
noun: p (-n)			
	m	low	high
	n	low	high
	f		high
noun: p (other)			
	m	med	low
	n	med	low
	f	low	med

ending: nil   as in "ein"

correct for: <(nom acc) s n> <nom s m>

Expected case: Nominative
possibilities: <(nom acc) s n> <nom s (m n)> <(nom acc) s f>

noun: s	<(nom acc) s n>	<nom s (m n)>	<(nom acc) s f>
m	low	veryhigh	
n	veryhigh	low	
f	low	med	low

noun: p	<(nom acc) s n>	<nom s (m n)>	<(nom acc) p *>
m	low	low	med
n	low	low	med
f	low	low	med

Expected case: Accusative

noun: s
use nominative entries

noun: p
use nominative entries

Expected case: Dative

noun: s
use accusative entries, with diminished confidence

noun: p
use accusative entries, with diminished confidence

Appendix C

FULL CHART OF PROBABILITIES, INTENT-GUESSING METHOD 3

—Chart of P(intent[i]) for all case-number-gender combinations.
—Chart of P(form[j] | intent[i]) for all plausible endings, for all case-number-gender combinations.

Introductory comments:

All probabilities are referred to the actual expected slot. Because the expectation cannot be usefully described as a probability distribution, there is not a true conditional probability for intent; i.e. it is not correct to consider P(intent[i] | expectation[k]). The actual situation is quite close, however. For purposes of this analysis, the proper approach is to consider that the case-number-gender expectation specifies one page (in a tabulation of probabilities) on which the relevant intent[i] probabilities can be found. A separate page must exist for each possible case-number-gender combination. I make two assumptions which simplify the construction of the table:

1. The conditional probabilities, P(form[i] | intent[j]), for the various orthographic forms do not depend on the expectation. In other words, although P(intent[i]) depends on the expected values, P(form[j] | intent[i]) does not. For instance, if the student actually intends <acc s m>, the probability that he will use "-en" is the same regardless of whether that intent is correct. Therefore, only one set of form[j] probabilities, P(form[j] | intent[i]), is necessary.

2. The student's performance on each grammatical category will be independent; i.e., whether he can use the correct gender for a noun does not depend on the case in which it will be used. This assumption allows the intent probabilities to be constructed simply as a product of terms for each grammatical category. For example, to compute P(<acc s m>) with an expectation of <nom s m>, simply calculate the product of the terms for the individual grammatical categories.

— case: should be nom, intent is nom; use P(accnom)
— number: should be s, intent is s; use P(samenum)
— gender: should be m, intent is m; use P(samegen)

So P(<acc s m>) = P(accnom) * P(samenum) * P(samegen) for the expected <nom s m>.

The following is the complete chart of probabilities needed for the analysis: I have given both symbolic terms and ad-hoc estimates of reasona[illegible] numerical values.

P(intent) for specific values:

CASE:

Expected	Intended		
nominative	nom:	P(nomnom)	= .92
	acc:	P(accnom)	= .05
	dat:	P(datnom)	= .03
accusative	nom:	P(nomacc)	= .04
	acc:	P(accacc)	= .90
	dat:	P(datacc)	= .06
dative	nom:	P(nomdat)	= .05
	acc:	P(accdat)	= .01
	dat:	P(datdat)	= .85

GENDER:

intent same as expected:	P(samegen) = .7
intent different:	P(diffgen) = .15

NUMBER:

intent same as expected:	P(samenumb) = .85
intent different:	P(diffnumb) = .15

<u>P(form|intent) for case-gender-number combinations:</u>
(no gender distinction in plural)

INTENT	FORM & PROBABILITY		
nominative			
masc	er, .95		
neuter	es, .95		
fem	e, .95		
plural	e, .95		
accusative			
masc	en, .85	er, .10	em, .05
neuter	es, .95		
fem	e, .95		
plural	e, .98		
dative			
masc	em, .85	en, .10	er, .05
neuter	em, .90	en, .10	
fem	er, .95	e, .05	
plural	en, .85	em, .10	e, .05

(In a number of instances, I could not come up with a plausible wrong form for a particular line of the table, even though there is clearly a finite probability that the student will produce a wrong answer. It is for this reason that the sum of the probabilities is not always 1.0, to acknowledge the existence of an unspecified "other" form.)

Appendix D

Detailed Descritpion of Verb Inflection Analysis

Verb ending analysis includes three inflections: subject, verb stem, and verb ending. The analytic strategy must first determine the grammatical slots represented by each inflection; then, various comparisons can be made.

Stem:

The stem inflection is composed of the forms taken by the stem vowel and stem consonant. The characterization is based on the following scheme:

"I" - if the form is the same as the infinitive form of the verb;
"s23" - if the form is the same as that taken by the verb in 2nd & 3rd person present;
"*" - if the form is the same as the infinitive and the verb does not normally show a distinction in this category for s23;
"X" - otherwise.

Having arrived at a characterization on this scheme, the program follows a simple decision tree to arrive at a composite specification for the stem as a whole:

"*" - if both stem consonant and stem vowel are "*";
"I" - if both are "I", or one is "I" and other "*", or stem vowel is "I" and stem consonant "X";
"s23" - if at least one is "s23" and neither is "I";
"C" - (Confused) otherwise.

From the composite specification, the next step is a table reference to obtain the corresponding set of slots. Verb types are specified by single-bit numbers (in octal); if more than one type applies, the corresponding bits are "OR"ed together. For multiple possibilities other than type, separate slot terms are employed.

```
  * — <* * 7>   (all categories, verb types 1,2 and 4);
  I — <* p 77>, (plural, all verb types),
      <1 s 37>, (1st singular, all types except 40),
      <2 s 7>,  (2nd & 3rd person singular, verb types
      <3 s 7>        1, 2 and 4);
s23 — <1 s 40>, (1st person singular, type 40 / modals),
      <2 s 70>, (2nd & 3rd person singular, verb types
      <3 s 70>       10, 20 & 40 / all those that change).
```

ENDING:

The ending, if recognizeable, represents grammatical slots of the same composition as the stem. Again, multiple verb types are given as a union of bits. The only verb type distinction really present here is whether coverage includes modal auxillaries (type 40). The ending table is:

```
Null ending — <1 s 40>, <3 s 40>
"-e" — <1 s 37>
"-t" — <3 s 37>, <2 p 77>
"-en" — <1 p 77>, <3 p 77>
```

COMPARISON OPERATIONS:

The first comparison operation constructs the stem/ending specification. The program evaluates all pairwise comparisons between the set of stem slots and the set of ending slots. If the values of both person and number of the two slots match exactly, a resultant term is generated with those two values and an intersection figure for verb type. The latter is the logical "AND" of the two figures, representing a set intersection as the type numbers are associated with individual bits. For the resultant person and number terms, "*" is eliminated if possible. For example:

```
<2 s 70> with <2 s 77> gives <2 s 70>
              (person, number match;  70 "AND" 77 = 70)
<2 s 70> with <3 s 77> gives NIL
              (person mismatch)
<* * 7>  with <1 s 40> gives <1 s 0>
              (person okay — "*" matches anything, use
              non-* value as result.  Ditto for number.
              Type, 7 "AND" 40 = 0 [null] )
```

The last example shows a resultant slot which applies to a null verb type. It is carried along to introduce slight greater flexibility into the diagnosis.

The subject characteristics are reduced to a single slot, if necessary, by a quick check with the verb ending. Then three comparison terms are generated, matching the subject against the two verb characteristics independently and against the composite of both: subject/stem, subject/ending, and subject/stem/ending. The match criteria are the same as above: exact match for person and number. (No verb type is associated with the subject characteristic, so there is no screening on that term at this step.)

Based on the various composite characteristics, the program can make quite detailed error comments. Here are a few examples of output from the program:

——————— RUN 1 ——————— (abbreviated)

Description generated by human expert:
(STMT (SJ ICH) (VERB SEHEN))

SENTENCE INPUT BY STUDENT ..
ICH SEHE

NOW PROCEEDING WITH ANALYSIS

.... Verb ending check... OK!

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: VERYHIGH

.... Summary Comments
Clause type correct (FV-2)

———————END OF RUN ———————

——————— RUN 2 ——————— (abbreviated)

Description generated by human expert:
(STMT (SJ ICH) (VERB SEHEN))

SENTENCE INPUT BY STUDENT ..
ICH SIEH

.... VERB SEHEN .. APPEARS AS SIEH
VERB DOES NOT SHOW THIS IRREGULARITY Should be 20 is 40
ERROR IS IN STEM
ERROR IS IN ENDING Should be E is *T*

.... Summary Comments
Clause type correct (FV-2)

——————— END OF RUN ———————

———— RUN 3 ———— (abbreviated)

Description generated by human expert:
(STMT (SJ ICH) (VERB SEHEN))

SENTENCE INPUT BY STUDENT ..
ER SEHEN.

.... PN SUBSTITUTION.. Should be (1 NOM S * PER)
is ((3 NOM S M PER) (WORD ER))

.... VERB SEHEN .. APPEARS AS SEHEN
INFINITIVE FORM USED

.... Summary Comments
Clause type correct (FV-2)

———— END OF RUN ————

———— RUN 4 ———— (abbreviated)
[note change in expected sentence]

Description generated by human expert:
(STMT (SJ ER) (VERB SEHEN))

SENTENCE INPUT BY STUDENT ..
SIE SEHEN

.... PN SUBSTITUTION.. Should be (3 NOM S M PER)
is ((3 NOM P * PER) (WORD SIE)

.... Verb ending check... OK!

.... Summary Comments
Clause type correct (FV-2)

———— END OF RUN ————

The following table gives a fairly complete presentation of the final analysis algorithm: which kinds of errors are covered and how they are detected. To save space, I have employed a few abbreviations in the table:

SE – stem/ending characteristic slots
JS – subject/stem
JE – subject/ending
JSE – subject/stem/ending
SC – stem consonant
SV – stem vowel

(verb types associated with specific characteristics):
JStype, JEtype, JSEtype, Ctype (Correct type).

"bad XXtype" means that the logical "AND" of that type with the correct type yields zero.

ERROR TYPE	INDICATION	COMMENTS
general	JSE null, or JSEtype = 0	"verb form error"
wrong type verb	bad JSEtype	"missed an irregularity" or "extra irregularity"
stem error	JS null, or bad JStype	"error is in stem" if stem-spec = "C" then add "stem type 'confused', maybe improper formation"
	stem-spec = "C"	"stem confusion, though grammatically correct if assumed to be s23"
	SC-spec = "X"	"incorrect formation of SC"
	SV-spec = "X"	"incorrect formation of SV"
ending error	JE null, or bad JEtype	"error is in ending"
no errors	no error flag	"verb ending check .. OK"

Appendix E

Exact Flow of PHRASEDIVIDER, the Algorithm Which Determines Clause Type

The presentation of multiply-cascaded conditionals can be difficult to follow. I have chosen an informal scheme in which indentation is used to indicate the scope of the condition. Furthermore, the condition is given by merely stating the boolean condition, (which may be a single variable). I have tried to avoid if - then - else if - then - else type structures, rather repeating some of the condition each time; parentheses are sometimes used when a condition appears which the program actually expresses by the structure of the conditional rather than an explicit check.

The innermost of the conditions is followed by a colon; the text after that refers to the conclusions appropriate at that point.

PRELIMINARY PROCESSING:

The clause is divided into sections delimited by the verbs:

```
one verb:     "A" — elements before verb
              "B" — elements after verb
              "NODI" indicator set "true"

two verbs:    "A" — elements before first verb
              "B" — elements between two verbs
              "C" — elements after second verb
              "FVDI" indicator "true" if finite verb
                 preceeds dependent infinitive
              "DIFV" indicator "true" if dependent
                 infinitive preceeds finite verb

set "LA" to the number of elements in section "A"
                    (mnemonic: Length of A)
set "LB" and "LC" likewise (LC 0 if C is empty)
```

MAIN FLOW:

```
1. (FV-1 categories)
  LA = 0         (finite verb is very first element in clause)
    then:        Conclude FV-1 with confidence "very high"
                 (henceforth written "FV-1, "very high"")
                 Go to "check" step.

2. (FV-L categories....first a general, all-encompassing
                 conditional; then specific checks)
  DIFV                               (inverted verb order, or
  or NODI and LB = 0                 (finite verb only and at
                                        end of clause, or
  or FVDI and LB = LC = 0            (non-inverted verb order,
                                        but both verbs together
                                        at end of clause, or
  or Rel. Word present and           (special category for slightly
       NODI and LB = 1               (distorted forms that may
    or FVDI and LB = 0               (still be relative clauses
              and LC = 1
```

```
.-then-          (now disentangle various possibilites)
    Rel. Word missing
      DIFV
           LA = 1:          FV-L, "medium"
           LA > 1:          FV-L, "high"
      not DIFV
           LA = 1:          FV-2, "high"
           LA = 2:          FV-2, "medium"
           LA > 2:          FV-L, "medium"

    DIFV and Rel. Word or Question Word present
           LA = 1:          FV-L, "high"
           LA > 1:          FV-L, "veryhigh"

  (-or else-)
    Rel. Word present (NODI or FVDI)
           LA = 1:          FV-1, "high"
           LA > 1:          FV-1, "veryhigh"

    Question word present (NODI or FVDI, verbs at end)
           LA = 1:          FV-2, "high"
           LA > 1:          FV-L, "high"

    (no Rel. Word or Question Word, and)
     DIFV and LB = 0 and LC neq 0:    (possible DI in
                                       Front Field, FV-2)
           Add DI to Front Field members ("A")
           Increment LA by 1
           Go to step 3, FV-2 processing.

    otherwise  (DIFV, no Rel. Word, odd placement)
           LA = 1:          FV-L, "low"
           LA > 1:          FV-L, "med"

3.  (FV-2 forms)
      (no check, include everything not already claimed)

      conclude:  FV-2, "veryhigh"
```

CHECKING OPERATIONS (Quoted material here represents diagnostic messages)

End field check, check for either null or occupied by a single element which qualifies for that position. First decide which field—

NODI and type = FV-L: Check will be on "B" field
FVDI: Check will be on "C" field
DIFV AND TYPE = FV-2: Check wil be on "c" field

Let "E" be the End field ("B" or "C") and "LE" its length.

LE = 0: fine
LE = 1 and "E" element can be in end field: fine.

otherwise: Set "CERR" indicator ("C-error") for later reference; also,
if FVDI & LB = 0 Record "DI position error, not separated from FV")
else
if LB = 1 & "B" element is Subject & either type = FV-1 or type = FV-2 & Question Word is present: decrease confidence two steps, "E.F. error (English?)"
else if LE = 1: decrease confidence, "end field error"
else if FVDI & LB = 0: decrease confidence, "DI not separated"
otherwise: Record "not German" declare no confidence, "End-field too big"

GENERAL CHECKING, CONTINUED

```
if type = FV-L,
  DIFV & LB neq 0:        decrease confidence,
                                  "elements between DI and FV"
  FVDI:                   decrease confidence, "FV preceeds
                                  infinitive in FV-L"

if type = Fv-2,
  Rel. Word present:   decrease confidence, "Rel word present"
  check LA  (statement should have exactly one element
                 in front field);
       LA = 1:                      fine
       LA >1 & DIFV:                declare no confidence,
                                            "Front Field too big
                                            for DI in F.F. & FV-2"
       LA > 1 & not DIFV,           Front Field error indicator set
         LA =2 & not CERR:          decrease confidence, "two
                                            member front field"

         LA > 2 or CERR:            decrease confidence two steps,
                                            "F.F. overly large"

       LA > 1 and not DIFV
        and Subject in "A"
        and no Object in "A":       decrease confidence,
                                            "Possible English
                                            pattern in F.F. error"
```

SEPARABLE PREFIX CHECKS
(omitted if confidence level is "none")

```
  Prefix position confused:     decrease confidence
  type = FV-L
       Prefix detached:         decrease confidence
       Prefix attached:         increase confidence
  type = FV-2
       Prefix attached:         decrease confidence
       Prefix detached:         increase confidence
```

.... end of PHRASEDIVIDER ....

Appendix F

Examples of PHASEDIVIDER Operation

The following represents direct output from the two main word order functions, PHRASEDIVIDER and PHRASETYPEREPORT. To facilitate a demonstration using many different sentence forms, I created a short test program, which allows a shorthand specification of a sentence merely as a list of clause-element names. The particular abbreviations used are:

```
S - subject       O - object     M - verb modifier
FV - finite verb       DI - dependent infinitive
RW - relative word (subordinating conjunction,
                        relative pronoun..)
QW - question word
```

The following additional abbreviations are a little less transparent. They specify separable prefixes, and must indicate which verb is involved, as well as whether the prefix is really to be considered as attached (a difficulty created by the shorthand representation). The following list will be augmented by explanatory comments immediately preceeding each example in which a prefix abbreviation occurs.

```
FVPFX - finite verb prefix, separated
FVAPFX - finite verb prefix, attached
DIPFX - dependent infinite prefix, separated
DIAPFX - dependent infinitive prefix, attached.
```

A last note: some error messages attempt to identify the text of the offending word. Because of the shorthand used here, no text is available, so the symbol "TEXT" will appear instead.

....
SENTENCE: (S DI FV O M) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
No confidence in sentence type, probably confused
last straw was: F.F. too big for DI in F.F.— FV-2

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: F.F. too big for DI in F.F.— FV-2
Confidence in FVT setting is: NONE

PD VALUE: CONFUSED

....
SENTENCE: (S O FV) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: MED

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: TWO MEMBER FRONT FIELD.
Confidence in FVT setting is: HIGH
Due to: Possible English pattern in F.F. error
Confidence in FVT setting is: MED
.. other comments:
Front field error (FV-2): 2 elements in F.F.

.... Summary Comments
Clause type correct (FV-2)

PD VALUE: FV-2

....
SENTENCE: (S FV O) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: VERYHIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH

.... Summary Comments
Clause type correct (FV-2)

PD VALUE: FV-2

....
SENTENCE: (RW FV O M) EXPECTED TYPE: FV-L

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: HIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: REL-WORD PRESENT
Confidence in FVT setting is: HIGH

.... Summary Comments
Relative clause failure...
no REL-word, probably not correct intent

PD VALUE: FV-2

....
SENTENCE: (QW S O FV) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-L , level of confidence: HIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-L
Confidence in FVT setting is: HIGH

.... Summary Comments
Apparent shift to FV-L / REL form for WORD question

PD VALUE: FV-L

....
SENTENCE: (FV S M DI) EXPECTED TYPE: FV-1

.... Phrase division done, sentence type report:
Final verdict is FV-1 , level of confidence: VERYHIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-1
Confidence in FVT setting is: VERYHIGH

.... Summary Comments
Clause type correct (FV-1)

PD VALUE: FV-1

....
SENTENCE: (M S FV O M DI) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-2 ; level of confidence: MED

Play by Play account of ana
Due to: INITIAL HYPOTHESIS o FV-2
Confidence in FVT ERYHIGH
Due to: TWO MEMBER FRONT
Confidence in FVT setting is: HIGH
Due to: Possible English pattern in F.F. error
Confidence in FVT setting is: MED
... other comments:
Front field error (FV-2): 2 elements in F:F.

.... Summary Comments
Clause type correct (FV-2)

PD VALUE: FV-2

....
SENTENCE: (S FV DI O M M) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: HIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: DI not separated
Confidence in FVT setting is: HIGH
... other comments:
DI position error, not separated from FV

.... Summary Comments
Clause type correct (FV-2)

PD VALUE: FV-2

```
..........
SENTENCE:  (DI FV O M S)            EXPECTED TYPE:  FV-2

..........  Phrase division done, sentence type report:
       Final verdict is FV-2 , level of confidence: VERYHIGH

       Play by Play account of analysis:
       Due to: INITIAL HYPOTHESIS , FVT set to FV-2
               Confidence in FVT setting is: VERYHIGH

..........  Summary Comments
       Clause type correct (FV-2)

PD VALUE: FV-2


 --- note:  in the following example, "FVPFX" denotes a
 ---    separable prefix associated with the finite verb

..........
SENTENCE: (S FV O M FVPFX)          EXPECTED TYPE:  FV-2

..........  Phrase division done, sentence type report:
       Final verdict is FV-2 , level of confidence: VERYHIGH

       Play by Play account of analysis:
       Due to: INITIAL HYPOTHESIS , FVT set to FV-2
               Confidence in FVT setting is: VERYHIGH
       Due to: SEPARABLE-PREFIX IS DETACHED
               Confidence in FVT setting is: VERYHIGH

..........  Summary Comments
       Clause type correct (FV-2)
       Prefix separated, position is Correct

PD VALUE: FV-2
```

— note: in the following example, "FVAPFX" denotes a
— separable prefix associated with the finite verb.
— It should be considered as not separated, even though
— here written separately.

....
SENTENCE: (S O FVAPFX FV). EXPECTED TYPE: FV-L

.... Phrase division done, sentence type report:
Final verdict is FV 2 , level of confidence: LOW

Play by Play account analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: TWO MEMBER FRONT FIELD.
Confidence in FVT setting is: HIGH
Due to: Possible English pattern in F.F. error
Confidence in FVT setting is: MED
Due to: SEPARABLE-PREFIX NOT DETACHED
Confidence in FVT setting is: LOW
.. other comments:
Front field error (FV-2): 2 elements in F.F.

.... Summary Comments
Relative clause failure...
no REL-word, probably not correct intent

.... Error in finite verb separable prefix, must be
separated and at end of clause in FV-2 clause

PD VALUE: FV-2

— note: in the following example, "FVPFX" denotes a
— separable prefix associated with the finite verb

....
SENTENCE: (S O FV FVPFX) EXPECTED TYPE: FV-L

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: HIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Cor[illegible]ence in FVT setting is: VERYHIGH
D[illegible]MBER FRONT FIELD.
Con[illegible]dence in FVT setting is: HIGH
Due to: Possible English pattern in F.F. error
Confidence in FVT setting is: MED
Due to: SEPARABLE-PREFIX IS DETACHED
Confidence in FVT setting is: HIGH
.. other comments:
Front field error (FV-2): 2 elements in F.F.

.... Summary Comments
Relative clause failure...
no REL-word, probably not correct intent.
Prefix separated, position is Correct

PD VALUE: FV-2

— note: in the following example, "DIPFX" denotes a
— separable prefix associated with the dependent
— infinitive.

....
SENTENCE: (S FV O DIPFX M DI) EXPECTED TYPE: FV-2

.... Phrase division done, sentence type report:
Final verdict is FV-2 , level of confidence: HIGH

Play by Play account of analysis:
Due to: INITIAL HYPOTHESIS , FVT set to FV-2
Confidence in FVT setting is: VERYHIGH
Due to: Separable-prefix confusion
Confidence in FVT setting is: HIGH

.... Summary Comments
Clause type correct (FV-2)
Prefix check for verb "TEXT" (DINF) FOLLOWS:
Position wrong (confused?) .. separated and before verb

PD VALUE: FV-2

```
—— note: in the following example, "DIPFX" denotes a
——   separable prefix associated with the dependent
——   infinitive.

..........
SENTENCE:  (S FV O DIPFX DI)      EXPECTED TYPE:  FV-2

..........  Phrase division done, sentence type report:
        Final verdict is FV-2 , level of confidence: HIGH

        Play by Play account of analysis:
        Due to: INITIAL HYPOTHESIS , FVT set to FV-2
                Confidence in FVT setting is: VERYHIGH
        Due to: Separable-prefix confusion
                Confidence in FVT setting is: HIGH

..........  Summary Comments
        Clause type correct (FV-2)
        Prefix check for verb "TEXT" (DINF) FOLLOWS:
        Position maybe okay..
                (immed. before verb, but separate word

PD VALUE: FV-2


—— note: in the following example, "FVPFX" denotes a
——   separable prefix associated with the finite verb

..........
SENTENCE:  (RW O M FV DI FVPFX)  EXPECTED TYPE:  FV-L

..........  Phrase division done, sentence type report:
        Final verdict is FV-L , level of confidence: MED

        Play by Play account of analysis:
        Due to: INITIAL HYPOTHESIS , FVT set to FV-L
                Confidence in FVT setting is: VERYHIGH
        Due to: FV preceeds DI in FV-L clause.
                Confidence in FVT setting is: HIGH
        Due to: PREFIX IS SEPARATED
                Confidence in FVT setting is: MED

..........  Summary Comments
        Correct verb position for relative clause

..........  Error in prefix, must not be separated when
                verb is in fv-1 position

PD VALUE: FV-L
```

```
—— note:  in the following example, "DIPFX" denotes a
——    separable prefix associated with the dependent
——    infinitive.


SENTENCE:  (RW O M DIPFX.DI FV)  EXPECTED TYPE:  FV-L

..........  Phrase division done, sentence type report:
        Final verdict is FV-L , level of confidence: HIGH

        Play by Play account of analysis:
        Due to: INITIAL HYPOTHESIS , FVT set to FV-L
                Confidence in FVT setting is: VERYHIGH
        Due to: Separable-prefix confusion
                Confidence in FVT setting is: HIGH

..........  Summary Comments
        Correct verb position for relative clause
        Prefix check for verb "TEXT" (DINF) FOLLOWS:
        Position maybe okay..
                (immed. before verb, but separate word

PD VALUE: FV-L
```

REFERENCES

(note: this section contains a listing of all references made within the main text, giving both the page cited in the source and a cross-reference to the chapter of this report on which the citation occurs.)

1. Adams, E.N., Morrison, H.W., and Reddy, J.M. "Conversation with a Computer as a Technique of Language Instruction," in Atkinson, R.C., and Wilson, H.A. (ed.) Computer-Assisted Instruction: A Book of Readings. New York: Academic Press, 1969, pages 173-190. (Reprinted from The Modern Language Journal, Vol. 1 No. 1 (Jan 1968), pp. 3-16)
   1.1: Entire; in Chapter 2
   1.2: p. 185; in Chapter 2

2. Carbonell, Jaime R. Mixed-Initiative Man-Computer Instructional Dialogues. Bolt, Beranek and Newman, Inc., Report No. 1971, 1970.
   2.1: Entire; in Chapter 2
   2.2: p. 86; in Chapter 2
   2.3: p. 185; in Chapter 2
   2.4: p. 88; in Chapter 2
   2.5: p. 87ff; in Chapter 2

3. Carbonell, Jaime R. Private Communication, 1972.
   3 : in Chapter 2

4. Corder, S.D. "The Signficance of Learners' Errors," International Review of Applied Linguistics in Language Teaching (IRAL), IV (Nov 1967).
   4.1: p. 162f; in Chapter 6
   4.2: p. 163; in Chapter 6

5. Corder, S.D. "Idiosyncratic Dialects and Error Analysis," IRAL IX (May 1971).
   5 : p. 147; in Chapter 6

6. Halliday, M.A.K., McIntosh, Angus, and Strevens, Peter. The Linguistic Sciences and Language Teaching. London: Longmans, 1964.
6 : p. 119; in Chapter 7

7. Kufner, Herbert L. The Grammatical Structures of English and German: A Contrastive Sketch. Chicago: University of Chicago Press, 196[illegible]
7.1: p. 54; in Chapter /
7.2: p. 9; in Chapter 8
7.3: p. 26; in Chapter 10
7.4: Entire; in Appendix A
7.5: p. 11; in Appendix 1

8. Lado, Robert. Language Teaching: a Scientific Approach. New York: McGraw-Hill, Inc., 1964.
8.1: p. 67; in Chapter 4
8.2: p. 80; in Chapter 6

9. Lederer, Herbert. Reference Grammar of the German Language. New York: Charles Scribner's Sons, 1969. Based on, and largely translated from: Schulz, Dora, and Griesbach, Heinz, Grammatik der Deutschen sprache. Munich: Max Hueber Verlag, 1965.
9.1: Entire; in Chapter 2
9.2: sec. C101; in Chapter 7
9.3: sec. B786; in Chapter 8
9.4: Entire; in Appendix A
9.5: sec. H110; in Appendix A
9.6: sec. H002; in Appendix A
9.7: sec. H022; in Appendix A
9.8: sec. H051; in Appendix A
9.9: sec. H233; in Appendix A
9.10: sec. H23[illegible]; in Appendix A

10. Lohnes, Walter F.W., and Strothman, F.W. German: A Structural Approach. New York: W.W. Norton & Co., Inc., 1968.
10.1: Entire; in Chapter 2
10.2: p. 143; in Chapter 4

11. Politzer, Robert L. Teaching French: An Introduction to Applied Linguistics. Boston: Ginn, 1960.
11 : p. 14; in Chapter 6
(also cited in Rivers <ref. 13>, p. 64)

12. Rehder, Helmut, and Twadell, Freeman. German.
New York: Holt, Rinehart & Winston, Inc., 1958.
12.1: p. 37; in Chapter 4
12.2: p. vi; in Chapter 4

13. Rivers, Wilga. The Psychologist and the Language Teacher. Chicago: University of Chicago Press, 1964.
13 : p. 61; in Chapter 6

14. Scholl, Pierre-Claude, and Scholl, Maryse. Forthcoming technical report on an instructional project in the French language. Institute for Mathematical Studies in the Social Sciences, Stanford University, 1972.
14.1: Entire; in Chapter 2
14.3: Private communication; in Chapter 2
14.3: Private communication; in Chapter 10

15. Siklossy, Laurent. "Computer Tutors That Know What They Teach," in Proceedings of the Fall Joint Computer Conference, 1970. Montvale, New Jersey: AFIPS Press, 1970
15 : Entire; in Chapter 2

16. Stake, Robert, "Comments on Prof. Glaser's paper" ("Evaluation of Instruction and Changing Educational Models"), in Wittrock, M.C., and Wiley, David (eds.). The Evaluation of Instruction: Issues and Problems. (Proceedings of a Symposium of Problems in the Evaluation of Instruction, UCLA, 1967). New York: Holt, Reinhart & Winston, 1970.
16 : p. 90; in Chapter 10

17. Suppes, Patrick, and Morningstar, Mona. "Technologcial Innovations: Computer-Assisted Instruction and Compensatory Education," in Korten, F., Cook, S., and Lacey, J. (eds.). Psychology and the Problems of Society. Washington, D.C.: American Psychological Association, Inc., 1970.
17 : p. 232ff; in Chapter 2

18. Symonds, Martha. Computer Detection of Misspelled Words. Report No. ERB-843. Ottawa: National Research Council of Canada, Radio and Electrical Engineering Division, 1970.
18 : Entire; in Chapter 10

19. Vail, Van Horn, and Sparks, Kimberly. (eds.) Der Weg zum Lesen: A German Structural Reader. New York: Harcourt, Brace & World, Inc., 1967.
19 : p. viii (introduction); in Chapter 4

20. Webster's Seventh New Collegiate Dictionary. Springfield, Mass.: G. & C. Merriam Co., 1965.
20 : "Inflection"; in Chapter 7

21. Wexler, Jonathan D. A Generative, Remedial and Query System for Teaching by Computer. Ph.D. dissertation, and report UWSCD-UTD-1, Computer Sciences Department, University of Wisconsin (Madison), 1970.
21.1: Entire; in Chapter 2
21.2: p. 85ff and p. 140ff; in Chapter 2

22. Winograd, Terry. Procedures as a Representation for Data in a Computer Program for Understanding Natural Language. Report MAC TR-84(Thesis), Project MAC, Massachusetts Institute of Technology, 1971.
22.1: Entire; in Chapter 2
22.2: pp. 9-15 or p. 89; in Chapter 2
22.3: p. 239ff (for Planner); in Chapter 5

FILMED FROM BEST AVAILABLE COPY

(Continued from inside front cover)

96 R. C. Atkinson, J. W. Breisford, and R. M. Shiffrin. Multi-process models for memory with applications to a continuous presentation task. April 13, 1966. (J. math. Psychol., 1967, 4, 277-300).
97 P. Suppes and E. Crothers. Some remarks on stimulus-response theories of language learning. June 12, 1966.
98 R. Bjork. All-or-none subprocesses in the learning of complex sequences. (J. math. Psychol., 1968, 1, 182-195).
99 E. Gammon. The statistical determination of linguistic units. July 1, 1966.
100 P. Suppes, L. Hyman, and M. Jerman. Linear structural models for response and latency performance in arithmetic. On J. P. Hill (ed.), Minnesota Symposia on Child Psychology. Minneapolis, Minn.: 1967. Pp. 160-200).
101 J. L. Young. Effects of intervals between reinforcements and test trials in paired-associate learning. August 1, 1966.
102 H. A. Wilson. An investigation of linguistic unit size in memory processes. August 3, 1966.
103 J. T. Townsend. Choice behavior in a cued-recognition task. August 8, 1966.
104 W. H. Batchelder. A mathematical analysis of multi-level verbal learning. August 9, 1966.
105 H. A. Taylor. The observing response in a cued psychophysical task. August 10, 1966.
106 R. A. Bjork. Learning and short-term retention of paired associates in relation to specific sequences of interpresentation intervals. August 11, 1966.
107 R. C. Atkinson and R. M. Shiffrin. Some Two-process models for memory. September 30, 1966.
108 P. Suppes and C. Ihrke. Accelerated program in elementary-school mathematics--the third year. January 30, 1967.
109 P. Suppes and I. Rosenthal-Hill. Concept formation by kindergarten children in a card-sorting task. February 27, 1967.
110 R. C. Atkinson and R. M. Shiffrin. Human memory: a proposed system and its control processes. March 21, 1967.
111 Theodore S. Rodgers. Linguistic considerations in the design of the Stanford computer-based curriculum in initial reading. June 1, 1967.
112 Jack M. Knutson. Spelling drills using a computer-assisted instructional system. June 30, 1967.
113 R. C. Atkinson. Instruction in initial reading under computer control: the Stanford Project. July 14, 1967.
114 J. W. Breisford, Jr. and R. C. Atkinson. Recall of paired-associates as a function of overt and covert rehearsal procedures. July 21, 1967.
115 J. H. Stelzer. Some results concerning subjective probability structures with semiorders. August 1, 1967
116 D. E. Rumelhart. The effects of interpresentation intervals on performance in a continuous paired-associate task. August 11, 1967.
117 E. J. Fishman, L. Keller, and R. E. Atkinson. Massed vs. distributed practice in computerized spelling drills. August 18, 1967.
118 G. J. Groen. An investigation of some counting algorithms for simple addition problems. August 21, 1967.
119 H. A. Wilson and R. C. Atkinson. Computer-based instruction in initial reading: a progress report on the Stanford Project. August 25, 1967.
120 F. S. Roberts and P. Suppes. Some problems in the geometry of visual perception. August 31, 1967. (Synthese, 1967, 17, 173-201)
121 D. Jamison. Bayesian decisions under total and partial ignorance. D. Jamison and J. Kozielecki. Subjective probabilities under total uncertainty. September 4, 1967.
122 R. C. Atkinson. Computerized instruction and the learning process. September 15, 1967.
123 W. K. Estes. Outline of a theory of punishment. October 1, 1967.
124 T. S. Rodgers. Measuring vocabulary difficulty: An analysis of item variables in learning Russian-English and Japanese-English vocabulary parts. December 18, 1967.
125 W. K. Estes. Reinforcement in human learning. December 20, 1967.
126 G. L. Wolford, D. L. Wessel, W. K. Estes. Further evidence concerning scanning and sampling assumptions of visual detection models. January 31, 1968.
127 R. C. Atkinson and R. M. Shiffrin. Some speculations on storage and retrieval processes in long-term memory. February 2, 1968.
128 John Holmgren. Visual detection with imperfect recognition. March 29, 1968.
129 Lucille B. Mlodnosky. The Frostig and the Bender Gestalt as predictors of reading achievement. April 12, 1968.
130 P. Suppes. Some theoretical models for mathematics learning. April 15, 1968. (Journal of Research and Development in Education, 1967, 1, 5-22)
131 G. M. Olson. Learning and retention in a continuous recognition task. May 15, 1968.
132 Ruth Norene Hartley. An investigation of list types and cues to facilitate initial reading vocabulary acquisition. May 29, 1968.
133 P. Suppes. Stimulus-response theory of finite automata. June 19, 1968.
134 N. Moler and P. Suppes. Quantifier-free axioms for constructive plane geometry. June 20, 1968. (In J. C. H. Gerretsen and F. Oort (Eds.), Compositio Mathematica. Vol. 20. Groningen, The Netherlands: Wolters-Noordhoff, 1968. Pp. 143-152.)
135 W. K. Estes and D. P. Horst. Latency as a function of number or response alternatives in paired-associate learning. July 1, 1968.
136 M. Schlag-Rey and P. Suppes. High-order dimensions in concept identification. July 2, 1968. (Psychom. Sci., 1968, 11, 141-142)
137 R. M. Shiffrin. Search and retrieval processes in long-term memory. August 15, 1968.
138 R. D. Freund, G. R. Loftus, and R.C. Atkinson. Applications of multiprocess models for memory to continuous recognition tasks. December 18, 1968.
139 R. C. Atkinson. Information delay in human learning. December 18, 1968.
140 R. C. Atkinson, J. E. Holmgren, and J. F. Juola. Processing time as influenced by the number of elements in the visual display. March 14, 1969.
141 P. Suppes, E. F. Loftus, and M. Jerman. Problem-solving on a computer-based teletype. March 25, 1969.
142 P. Suppes and Mona Morningstar. Evaluation of three computer-assisted instruction programs. May 2, 1969.
143 P. Suppes. On the problems of using mathematics in the development of the social sciences. May 12, 1969.
144 Z. Domotor. Probabilistic relational structures and their applications. May 14, 1969.
145 R. C. Atkinson and T. D. Wickens. Human memory and the concept of reinforcement. May 20, 1969.
146 R. J. Titiev. Some model-theoretic results in measurement theory. May 22, 1969.
147 P. Suppes. Measurement: Problems of theory and application. June 12, 1969.
148 P. Suppes and C. Ihrke. Accelerated program in elementary-school mathematics--the fourth year. August 7, 1969.
D. Rundus and R.C. Atkinson. Rehearsal in free recall: A procedure for direct observation. August 12, 1969.
P. Suppes and S. Feldman. Young children's comprehension of logical connectives. October 15, 1969.

(Continued on back cover)

(Continued from inside back cover)

151 Joaquim H. Laubsch. An adaptive teaching system for optimal item allocation. November 14, 1969.
152 Roberta L. Klatzky and Richard C. Atkinson. Memory scans based on alternative test stimulus representations. November 25, 1969.
153 John E. Holmgren. Response latency as an indicant of information processing in visual search tasks. March 16, 1970.
154 Patrick Suppes. Probabilistic grammars for natural languages. May 15, 1970.
155 E. Gammon. A syntactical analysis of some first-grade readers. June 22, 1970.
156 Kenneth N. Wexler. An automaton analysis of the learning of a miniature system of Japanese. July 24, 1970.
157 R. C. Atkinson and J.A. Paulson. An approach to the psychology of instruction. August 14, 1970.
158 R.C. Atkinson, J.D. Fletcher, H.C. Chetin, and C.M. Stauffer. Instruction in initial reading under computer control: the Stanford project. August 13, 1970.
159 Dewey J. Rundus. An analysis of rehearsal processes in free recall. August 21, 1970.
160 R.L. Klatzky, J.F. Juola, and R.C. Atkinson. Test stimulus representation and experimental context effects in memory scanning.
161 William A. Rottmayer. A formal theory of perception. November 13, 1970.
162 Elizabeth Jane Fishman Loftus. An analysis of the structural variables that determine problem-solving difficulty on a computer-based teletype. December 18, 1970.
163 Joseph A. Van Campen. Towards the automatic generation of programmed foreign-language instructional materials. January 11, 1971.
164 Jamesine Friend and R.C. Atkinson. Computer-assisted instruction in programming: AID. January 25, 1971.
165 Lawrence James Hubert. A formal model for the perceptual processing of geometric configurations. February 19, 1971.
166 J. F. Juola, I.S. Fischler, C.T. Wood, and R.C. Atkinson. Recognition time for information stored in long-term memory.
167 R.L. Klatzky and R.C. Atkinson. Specialization of the cerebral hemispheres in scanning for information in short-term memory.
168 J.D. Fletcher and R.C. Atkinson. An evaluation of the Stanford CAI program in initial reading (grades K through 3). March 12, 1971.
169 James F. Juola and R.C. Atkinson. Memory scanning for words versus categories.
170 Ira S. Fischler and James F. Juola. Effects of repeated tests on recognition time for information in long-term memory.
171 Patrick Suppes. Semantics of context-free fragments of natural languages. March 30, 1971.
172 Jamesine Friend. Instruct coders' manual. May 1, 1971.
173 R.C. Atkinson and R.M. Shiffrin. The control processes of short-term memory. April 19, 1971.
174 Patrick Suppes. Computer-assisted instruction at Stanford. May 19, 1971.
175 D. Jamison, J.D. Fletcher, P. Suppes, and R.C. Atkinson. Cost and performance of computer-assisted instruction for compensatory education.
176 Joseph Offir. Some mathematical models of individual differences in learning and performance. June 28, 1971.
177 Richard C. Atkinson and James F. Juola. Factors influencing speed and accuracy of word recognition. August 12, 1971.
178 P. Suppes, A. Goldberg, G. Kanz, B. Searle, and C. Stauffer. Teacher's handbook for CAI courses. September 1, 1971.
179 Adele Goldberg. A generalized instructional system for elementary mathematical logic. October 11, 1971.
180 Max Jerman. Instruction in problem solving and an analysis of structural variables that contribute to problem-solving difficulty. November 12, 19
181 Patrick Suppes. On the grammar and model-theoretic semantics of children's noun phrases. November 29, 1971.
182 Georg Kreisel. Five notes on the application of proof theory to computer science. December 10, 1971.
183 James Michael Moloney. An investigation of college student performance on a logic curriculum in a computer-assisted instruction setting. January 28, 1972.
184 J.E. Friend, J.D. Fletcher, and R.C. Atkinson. Student performance in computer-assisted instruction in programming. May 10, 1972.
185 Robert Lawrence Smith, Jr. The syntax and semantics of ERICA. June 14, 1972.
186 Adele Goldberg and Patrick Suppes. A computer-assisted instruction program for exercises on finding axioms. June 23, 1972.
187 Richard C. Atkinson. Ingredients for a theory of instruction. June 26, 1972.
188 John D. Bonvillian and Veda R. Charrow. Psycholinguistic implications of deafness: A review. July 14, 1972.
189 Phipps Arabie and Scott A. Boorman. Multidimensional scaling of measures of distance between partitions. July 26, 1972.
190 John Ball and Dean Jamison. Computer-assisted instruction for dispersed populations: System cost models. September 15, 1972.
191 W. R. Sanders and J. R. Ball. Logic documentation standard for the Institute for Mathematical Studies in the Social Sciences. October 4, 1972.
192 M.T. Kane. Variability in the proof behavior of college students in a CAI course in logic as a function of problem characteristics. October 6, 1972.
193 P. Suppes. Facts and fantasies of education. October 18, 1972.
194 R. C. Atkinson and J. F. Juola. Search and decision processes in recognition memory. October 27, 1972.
195 P. Suppes, R. Smith, and M. Léveillé. The French syntax and semantics of PHILIPPE, part 1 : Noun phrases. November 3, 1972.
196 D. Jamison, P. Suppes, and S. Wells. The effectiveness of alternative instructional methods: A survey. November 1972.
197 P. Suppes. A survey of cognition in handicapped children. December 29, 1972.
198 B. Searle, P. Lorton, Jr., A. Goldberg, P. Suppes, N. Ledet, and C. Jones. Computer-assisted instruction program: Tennessee State Universit February 14, 1973.
199 D. R. Levine. Computer-based analytic grading for German grammar instruction. March 16, 1973.